CANADIAN THESES ON MICROFICHE

THÈSES CANADIENNES SUR MICROFICHE

National Library of Canada
Collections Development Branch

Canadian Theses on
Microfiche Service

Ottawa, Canada
K1A 0N4

Bibliothèque nationale du Canada
Direction du développement des collections

Service des thèses canadiennes
sur microfiche

NOTICE

The quality of this microfiche is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Previously copyrighted materials (journal articles, published tests, etc.) are not filmed.

Reproduction in full or in part of this film is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30. Please read the authorization forms which accompany this thesis.

AVIS

La qualité de cette microfiche dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

Les documents qui font déjà l'objet d'un droit d'auteur (articles de revue, examens publiés, etc.) ne sont pas microfilmés.

La reproduction, même partielle, de ce microfilm est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30. Veuillez prendre connaissance des formules d'autorisation qui accompagnent cette thèse.

THIS DISSERTATION
HAS BEEN MICROFILMED
EXACTLY AS RECEIVED

LA THÈSE A ÉTÉ
MICROFILMÉE TELLE QUE
NOUS L'AVONS REÇUE

Canada

NL 339 (r. 85/09)

0-315-21207-1

National Library of Canada
Canadian Theses Division
Ottawa, Canada
K1A 0N4

Bibliothèque nationale du Canada
Division des thèses canadiennes

(11)

PERMISSION TO MICROFILM — AUTORISATION DE MICROFILMER

• Please print or type — Écrire en lettres moulées ou dactylographier

Full Name of Author — Nom complet de l'auteur

MAZUMDER PINAKI

Date of Birth — Date de naissance

JULY 04, 1954.

Country of Birth — Lieu de naissance

INDIA.

Permanent Address — Résidence fixe

38, Narendra Nagar.
Calcutta - 700 056.
W.B., India.

Title of Thesis — Titre de la thèse

Networks & Embedding Aspects of Cellular Structures for On-chip Parallel Processing in VLSI.

University — Université

Univ. of Alberta.

Degree for which thesis was presented — Grade pour lequel cette thèse fut présentée

M.Sc.

Year this degree conferred — Année d'obtention de ce grade

1985 (Spring)

Name of Supervisor — Nom du directeur de thèse

Prof J. Tartar.

Permission is hereby granted to the NATIONAL LIBRARY OF CANADA to microfilm this thesis and to lend or sell copies of the film.

The author reserves other publication rights, and neither the thesis nor extensive extracts from it may be printed or otherwise reproduced without the author's written permission.

L'autorisation est, par la présente, accordée à la BIBLIOTHÈQUE NATIONALE DU CANADA de microfilmer cette thèse et de prêter ou de vendre des exemplaires du film.

L'auteur se réserve les autres droits de publication; ni la thèse ni de longs extraits de celle-ci ne doivent être imprimés ou autrement reproduits sans l'autorisation écrite de l'auteur.

Date

Signature

NL-91 (4/77)

The University of Alberta

NETWORKS AND EMBEDDING ASPECTS OF CELLULAR STRUCTURES FOR ON-CHIP PARALLEL PROCESSING IN VLSI

by

Pinaki Mazumder

A thesis
submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree
of Master of Science

Department of Computing Science

Edmonton, Alberta
Spring, 1985

THE UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR: Pinaki Mazumder

TITLE OF THESIS: Networks and Embedding Aspects of Cellular Structures for On-chip Parallel Processing in VLSI

DEGREE FOR WHICH THIS THESIS WAS PRESENTED: Master of Science

YEAR THIS DEGREE GRANTED: 1985

Permission is hereby granted to The University of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves other publication rights, and neither the thesis nor extensive extracts from it may be printed or otherwise reproduced without the author's written permission.

(Signed) ..... 31/10/84 .....

Permanent Address:

38, Narendra Nagar
Calcutta 700 056
INDIA

Dated 31 October 1984

THE UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research, for acceptance, a thesis entitled **Networks and Embedding Aspects of Cellular Structures for On-chip Parallel Processing in VLSI** submitted by **Pinaki Mazumder** in partial fulfillment of the requirements for the degree of **Master of Science.**

.....
Supervisor

.....

.....

.....

Date November 1, 1984

ABSTRACT

The advent of VLSI has opened a new vista for parallel processing. On-chip parallel processing by numerous tiny processors is envisioned to be the major application goal of the emerging VLSI and VHSIC technologies.

This thesis attempts to identify the interconnection networks and the processor geometries which can be cost-effectively implemented within the chip for on-chip parallel processing. The thesis has proposed a computational model for CMOS VLSI technology and has evaluated the existing SIMD interconnection networks employing the results of the model. The interconnection networks have been divided into four topologically equivalent classes and the merits and demerits of each class have been perused to reveal the orthogonal aspects of parallel computation viz., the physical aspects, the computational aspects and the reliability aspects. Cellular interconnection networks having uniform and short interconnects have been shown to be the most suitable candidate for on-chip parallel processing.

The layout geometry for such networks has been investigated. The traditional shape for the processors is understood to be a square. The algorithms for transforming a natural layout having a rectangular shape with arbitrary aspect ratio into a square layout have been constructed. The difficulties and disadvantages with such post-processing have been studied and an alternate strategy has been suggested. A number of possible geometrical shapes for the processors have been identified.

Mosaic layout by cellular structures having a cell geometry identical to polyomino tiles has been designed for the square and the hexagonal array networks. Polyomino tiles which describe the fault-tolerant mesh networks have been identified. Embedding algorithms have been designed to construct such networks with optimal redundancy. Finally, the chip area, the layout cost and the computational power of such layout structures have been analyzed.

Acknowledgements

I gratefully acknowledge the assistance of:

Professor J. Tartar, my thesis supervisor, for countless helpful comments and discussions. Working with him was a pleasure and a privilege.

Professors S. Cabay, T. Marsland and J. Mowchenko who have examined my thesis. Prof. K. Stromsmoe for teaching me VLSI technology.

Swami Vikashanandaji, my father Shri Animesh C. Mazumder and my mother Shrimati C. Mazumder who have instilled in me the ambition to pursue for higher studies. This thesis is dedicated to them.

My wife Deepika, for her patience and her moral support.

Members of the faculty and staff of the Department of Computing Science of the University of Alberta.

Table of Contents

	Page
Chapter 1 INTRODUCTION	1
1.1 Introduction:	1
1.2 Objectives of the thesis:	2
1.2.1 Problem to be addressed:	2
1.2.2 Objectives and Methods:	3
1.3 Review of Related Work and Justification	3
1.4 Overview of the Thesis	7
Chapter 2 VLSI MODEL OF COMPUTATION	9
2.1 Introduction	9
2.2 Growth complexity of an IC with time	9
2.3 Rudiments of an Abstract Model	10
2.4 Computational Model:	11
2.4.1 Technological Assumptions:-	12
2.4.2 Embedding Assumptions:-	12
2.4.3 Timing Assumptions:-	14
2.4.4 Energy Dissipation Assumption:-	15
2.4.5 Failure Assumptions:-	19
2.4.6 Critical Appraisal of the Model:	23
2.4.7 Conclusion:	25
Chapter 3 EVALUATION OF NETWORKS	26
3.1 Introduction:	26
3.2 Classification of Networks:	26
3.2.1 Identification of networks for evaluation	28

3.3 Criteria for Evaluation: 28
3.3.1 Physical aspects: 29
3.3.1.1 Chip Area 29
3.3.1.2 Power Consumption 29
3.3.2 Computational Aspects: 29
3.3.2.1 Delay 29
3.3.2.2 Message Traffic Density 30
3.3.3 Cost Aspects: 30
3.3.3.1 Yield 30
3.3.3.2 Regularity 31
3.3.3.3 Fault-tolerance 31
3.4 Evaluation of Networks: 32
3.4.1 Two Dimensional Meshes 32
3.4.1.1 Discussion of Related Networks: 34
3.4.2 Binary Tree Networks: 35
3.4.2.1 Discussion of Related Networks: 39
3.4.3 Cube Connected Cycles: 40
3.4.3.1 Discussion of Related Networks 42
3.5 Comparison of Three Classes of Networks: 43
3.6 Conclusion: 43
Chapter 4 ALGORITHMS FOR LAYOUT TRANSFORMATION 45
4.1 Introduction: 45
4.2 Algorithms for layout transformation: 45
4.2.1 Break and Fold Embedding: 46

4.2.2 Step Embedding: 49
4.2.3 Embedding by Folding 50
4.3 Discussion of results: 54
4.4 An Alternate Layout Strategy: 54
4.5 Conclusion: 56
Chapter 5 CELLULAR EMBEDDING AND NETWORKS 57
5.1 Introduction 57
5.2 Formal Framework of Cellular Embedding 58
5.3 Mosaic Layout Constructed by Polyomino Tiles: 63
5.3.1 Embedding Algorithms 64
5.3.2 Computational Power 65
5.3.3 Chip Area 66
5.3.4 Layout cost 67
5.4 Interconnection Networks described by Mosaic Layout: 67
5.5 Cell geometries for Cellular Networks: 70
5.6 Conclusion: 71
Chapter 6 CONCLUSION 72
6.1 Summary: 72
6.2 Contribution of the thesis: 73
6.3 Recommendation of Future Research: 74
6.3.1 Extension of Computational Model: 74
6.3.2 Evaluation under Multi-level Interconnect Model: 74
6.3.3 Exploration of other Cellular Geometries: 75
Bibliography 76

List of Tables

	Page
1.1 Layout area of Networks	5
1.2 Area-time tradeoffs for different Networks	6
2.1 Comparison of Different Computational Models	25
3.1 Evaluation of Three Classes of Networks	44
5.1 Layout Geometries for Cellular Networks	71

List of Figures

	Page
2.1 Integrated Circuit complexity growth with time	10
2.2 Transmission line model of Interconnection Wire	15
2.3 Reduction of Delay in Polysilicon Wire by introducing Drivers	16
2.4 Reduction of Delay by introducing a Driver Chain	17
2.5 Occurence of Circular Defects of diameter η	20
2.6 Yield vs. Area for Defect Density, $D_0 = 9$ per cm^2	22
2.7 Yield vs. Redundancy for different Defect Density, D_0	23
3.1 A 4 x 4 Mesh Network	32
3.2 Layout of 4 x 4 Mesh Network	33
3.3 Mesh Network with Tree Interconnection	34
3.4 Layout of Binary Tree	35
3.5 The H-Tree Networks	36
3.6 Layout for H-Tree Networks	37
3.7 Message Flow through a level k node in a Tree Network	38
3.8 Layout of an X-Tree	39
3.9 Modified Binary Tree suitable for VLSI Implementation	40
3.10 Cube Connected Cycle topology for 3.2^3 processors	40
3.11 Layout of Cube Connected Cycle with 3.2^3 processors	41
4.1 Transformation of Layout by Break and Fold	46
4.2 Layout Transformation by Step Bending	49
4.3 Embedding by Folding	52

Page

5.1 Permissible Orientation of P_5 60

5.2 Shape Polynomial of Some Polyominoes 61

5.3 Planar Regions having Moore's Neighborhood 62

5.4 Mosaic Layout described by L_3 65

5.5 Communication Graphs described by Mosaic Layout 70

Chapter 1

INTRODUCTION

1.1. Introduction:

Parallel Processing Architecture has always played second fiddle to the state-of-the-art of semiconductor technology. The progress and development in the integrated circuit (IC) technology has continuously dictated the design philosophy of parallel architectures. The evolution of IC technology from Small Scale Integration to Large Scale Integration has chronologically motivated the design of connection schema of parallel architectures from simple structures like bus, star, ring, etc., to complex, powerful topologies like cube connected cycles, perfect shuffle network, dual bus hypercube, etc. An excellent account of this development has been presented in the literature [AnJ75, Thu74].

Now with the emergence of Very Large Scale Integrated (VLSI) circuit technology, parallel processing has entered into a new design phase. With the projected submicron technology of feature size of 0.625μ, more than 10^7 transistors can be integrated within a chip of area 100 mm^2 [Bar80]. This has opened the possibility to fabricate a large number of processors within a single IC. Algorithm based parallel processing is envisioned to be one of the potential application areas of the forthcoming submicron-level VLSI and Very High Speed Integrated Circuits (VHSIC) [Bar80]. Unlike the earlier parallel processing architectures using mini- or micro-computers as computing elements, parallel processors in VLSI compute within the chip under different constraints. A straightforward transcription of the interconnection topologies designed for inter-linking multi-microcomputer systems, does not ideally suit VLSI

implementation and a fresh evaluation of these topologies is needed to accommodate the new perspectives provided by the VLSI technology. This thesis addresses the network topologies and layout structures which are more suited for VLSI implementation.

1.2. Objectives of the thesis:

1.2.1. Problem to be addressed:

In order to justify the need for re-evaluation of interconnection networks, it should be noted that in the design of earlier networks adapted for non-VLSI environment

a. spatial distribution of the processors is not a constraint on the design,
b. signal propagation time is exclusively determined by the velocity of electromagnetic wave in the resistive medium and is negligibly small compared to the speed of operation. Thus the length of the interconnecting wire is not a constraint on design,
c. cost of the system is directly proportional to the average number of links per node,
d. fault-tolerance capability of such networks is merely a topolgical property (i.e., whether alternate message transmission routes exist or not)

On the contrary, under a VLSI environment

a. the spatial distribution of the processors play an important role on the total chip area. Additionally, the geometrical shape of the processor has direct bearing on the chip area,
b. the interconnecting wire behaves as transmission line having both resistive and capacitive components and signal propagation time is largely dependent on these values which are directly proportional to the length of the interconnection,
c. link per node does not have any direct relevance to design cost. The regularity of the networks topology and the processor geometry decide the layout cost,
d. fault-tolerance has an additional role to play. To improve the device yield and thereby to reduce the overall cost, it is necessary to introduce redundant processors. The network topology plays an important role in assigning the location of the redundant cells.

This thesis specifically addresses the following two problems:

(1) How these new constraints modify the performance of these interconnection networks;

(2) Given an interconnection network, what will be the optimal processor geometry which can be cost-effectively laid on the chip.

1.2.2. Objectives and Methods:

In order to contain the scope of the thesis and also to focus attention on the relevant aspects of VLSI technology, two conceptual simplifications have been made.

i) A theoretical model of VLSI computation has been proposed and the relevant electrical parameters of the VLSI technology has been analyzed asymptotically.

ii) The interconnection networks have been divided into three equivalent classes depending on their spatial distribution in 3-D space. The three classes are: networks which have spatial distribution on i) **planar** surface, ii) **spherical** surface and iii) **conical** surface. A representative of these three classes are **Mesh Network, Cube Connected Cycle** and **Binary Tree**, respectively.

The objective behind the performance evaluation of these three classes of networks is to prove that unlike the earlier networks under non-VLSI environment, regular structured cellular array networks with uniformly short interconnection length, is highly suitable for VLSI implementation.

The next objective is to identify the layout geometry of the processors that can be cost-effectively embedded on the chip to describe cellular networks. Specifically, the geometric structures which describe fault-tolerant mesh and hexagonal arrays will be identified. Logarithmic and polynomial layout algorithms will be designed employing the concept of a *linear transformation.*

1.3. Review of Related Work and Justification

To emphasize the motivation of this thesis, it is appropriate to enumerate briefly the past work in the related areas. The emergence of VLSI as an entity of theoretical computation is principally due to amalgamation of research work in a few apparently unrelated areas - the work of **Donath** [Don79], **Sutherland** and **Oestreicher** [SuO73]

in the area of automated PCB layout, **Cutler** and **Shiloach** [CuS78], **Lipton** and **Tarjan** [LiT80] and **Rosenberg** [AlR80] in the area of planar embeddings, **Moshell** and **Rothstein** [MoR76], **Abelson** [AbA80] and **Yao** [Yao79] in the area of information theoretic models for computational problems. Combining the ideas of the above work, **Thompson** [Tho80] has provided the basis of complexity theory for VLSI. The most significant contribution of his work is to correlate the chip area (A) with the speed of computation (T) for any arbitrary interconnection network and to suggest AT^2 as a performance metric of the parallel architectures. **Thompson** has defined the "minimal bisection width", w of a communication network as the smallest number of links between the processors which on removal reduce the original network into two equal-sized sub-networks. He has shown that the lower bound of the chip area, A can be given by $A \geq w^2/4$. **Leighton** [Lei81] has improved this lower bound to $A \geq c+N \geq w^2/4$ by introducing the notion of "crossing sequence", c. Crossing sequence of an N node planar graph is defined as the minimum number of pairs of edges which must cross in any planar embedding. Other notable work in the area of lower bound of chip area calculation is due to **Lipton** and **Sedgewick** [LiS81], **Yao** [Yao81], **Savage** [Sav81], **Brent** and **Goldschlager** [BrG82]. They introduce different techniques to obtain similar bounds.

The main practical problem with these theoretical results is that in many networks like the Perfect Shuffle Network it is not known how to attain this bound [StR80]. Also, the problem of finding the "minimum bisection width" in an arbitrary network is an NP-Complete problem [GJS74]. First practical layout techniques for an arbitrary network have been simultaneously (but independently) suggested by **Leiserson** [Lei81] and **Valiant** [Val81]. They have employed **Lipton** and **Tarjan's** [LiT80] *Planar Separator Theorem* to recursively partition the network into two equal-sized sub-networks by deleting $c_s f(n)$ links at every step. $c_s > 0$ is constant and $f(n)$ is the minimum number of processors that should be removed to separate the network into

two equal-sized subnetworks. Hence, it is called the $f(n)$-separator and is the property of the topology of the network. **Leiserson** [Lei81] and **Leighton** [Lei82] have given the analytical relationship on the upper bounds of the chip area of a network from its value of the $f(n)$-separator. Table 1.1 presents the upper and lower bounds of different interconnection networks employing one of the techniques described so far.

Another aspect of **Thompson's** work [Tho80] is to relate the speed of computation with the "minimum bisection width" of the network. If I is the minimum amount of information which must be transferred across w to solve an arbitrary computational problem, then **Thompson** has shown that the lower bound of T can be given as $T \geq (I/w)$ such that $AT^2 \geq I^2/4$. Utilizing this result, **Thompson** has calculated that an N point Discrete Fourier Transform can be implemented if and only if $AT^2 \geq ((N log N)^2)/16)$. Later on, the same metric has been used to arrive at the AT^2 value for other computational problems like bit-multiplication [AbA80], [BrK80], sorting [Tho83], etc. An extensive account of all these excellent theoretical works is available in [Ull84]. Table 1.2 highlights how this bound compares with different networks for an N point sorting problem. From the results of Table 1.2, it is evident that regular structured networks like a mesh is slower, but requires a small computational area.

Table 1.1 Layout area of Networks

Class of Networks	Lower Bound	Upper Bound
Binary Tree	$O(N)$	$O(N)$
Planar Network	$O(N)$	$O(N \log^2 N)$
k-dimensional meshes ($k>2$)	$O(N^{2-2/k})$	$O(N^{2-2/k})$
Cube Connected Cycles	$O(N^2/log^2 N)$	$O(N^2/log^2 N)$
Perfect Shuffle Network	$O(N^2/log^2 N)$	$O(N^2/log^{3/2} N)$

Fast networks like Perfect Shuffle Network, Cube Connected Cycle, etc., require more computational area.

Without being overcritical of these excellent theoretical results, it should be noted that there are some practical limitations to this work:

(1) The results do not take into account the propagation delay due to long wires,
(2) The results assume $O(1)$ area for all computational nodes irrespective of heterogeneous fan-in, fan-out and drive requirements.
(3) The AT^2 metric for different networks in Table 1.2 look to be identical leaving very little choice to prefer one network over another.
(4) Even the choice of AT^2 metric is under dispute. It is not convincingly known what will be the composite relationship indicating area and time trade offs. Many researchers [MeR82] consider that AT indicates the "rental time" of the chip for the computation of a problem and may be the correct metric. **Savage** [Sav81] has shown that for computational problems like binary sorting A^2T provides a better metric.
(5) Also, these results do not include the criteria for practical choice like fault-tolerance, layout cost, etc.

This thesis proposes to assess the interconnection networks from a more practical point of view. The approach adopted here can be described as "realistically asymptotic". The evaluation schema constitutes a hypothetical 3-D model - *area, time* and *cost*. It is well-understood that it is not possible to propose an analytical model

Table 1.2 Area-time tradeoffs for different Networks

Sorting Under The Constant Delay Model			
Network	Area	Time	Area×time^2
Two Dimensional Mesh	$N \log^2 N$	$N^{1/2}$	$N^2 \log^2 N$
Perfect Shuffle Network	$\frac{N^2}{\log^2 N}$	$\log^2 N$	$N^2 \log^2 N$
Cube Connected Cycles	$\frac{N^2}{\log^2 N}$	$\log^2 N$	$N^2 \log^2 N$
Tree of Meshes	$N^2 \log^2 N$	$\log N$	$N^2 \log^4 N$

combining all these parameters, because exact interaction of parameters like cost, area and time cannot be strictly generalized. But, overall, this provides a far clearer picture of network behavior in a VLSI environment.

1.4. Overview of the Thesis

The thesis can be broadly divided into three parts. The first part consists of Chapter 2 which gives a formal computational model of VLSI design. The second part of the thesis consists of Chapter 3 which explores the suitability of interconnection networks in a VLSI environment. The third and final part consists of Chapters 4 and 5, which addresses the layout optimality of Cellular Networks.

Specifically, Chapter 2 proposes a Formal Model of VLSI computation. The goal of this chapter is to propose a number of assumptions and theorems which asymptotically model the technological and electrical parameters. This has been done with a view to eliminating the obfuscating details of the actual device technology and simplifying the evaluation of the networks subsequently.

Chapter 3 evaluates the three classes of interconnection networks using the model described in the earlier chapter. The choice of the networks has been justified, the criteria for evaluation have been identified and the results of the evaluation have been compared to derive the conclusion that ***regular structured Cellular Networks are ideally suited for VLSI computation.***

Chapter 4 addresses the layout conversion problem. The objective behind introducing this chapter is to identify the layout geometry which describes close packing of the processors inside the IC. At first, the layout transformation algorithms have been proposed which can convert a rectangular layout of arbitrary aspect ratio to a square shape. The choice of the square shape has been due to the fact that, in general, they describe a very good packing density. The limitations of such techniques have been identified and alternate geometries have been proposed.

Chapter 5 utilizes the proposed geometries in the earlier chapters to construct the layout algorithms which describe the Cellular Networks. Two specific layout algorithms have been described which represent fault-tolerant meshes network and hexagonal arrays. The redundancy in fault-tolerant mesh has been shown to be near optimal. Two basic results of this chapter are:

(1) identification of the cell geometries which can be optimally laid out to construct a specific Cellular Network,
(2) identification of a class of Cellular Networks which can be constructed by polyomino tiles.

Chapter 6 derives the conclusion of the thesis. It enumerates the specific contributions of the thesis and the scope of future work.

Four specific contributions of this thesis are:

(1) a proposal of a formal computational model which represents all the aspects of CMOS VLSI technology,
(2) an identification of the Cellular Networks as most suitable candidate for parallel computation,
(3) an identification of processor geometries which can be cost-effectively laid on the chip to construct Cellular Networks,
(4) an identification of new class of Cellular Networks which are highly relevant in VLSI computation.

Chapter 2

VLSI MODEL OF COMPUTATION

2.1. Introduction

This chapter proposes an asymptotic, graph theoretic model of the integrated circuit. In order to simplify the evaluation of performance of the integrated circuit (IC), the electrical and technological parameters of the IC have been analyzed asymptotically. Detailed and rigorous analysis involving electrical circuits and device equations have been omitted and simple models have been proposed retaining only the relevant parameters. Section 2.2 enumerates the complexity of the growth of the integrated circuit with time and justifies the application of asymptotic analysis. Section 2.3 provides the rudiments of an abstract model and section 2.4 proposes the computational model that will be used in this thesis. Section 2.5 compares the proposed model with the existing models and section 2.6 concludes the chapter.

2.2. Growth complexity of an IC with time

Over the past two decades, the complexity of device integration in silicon technology has grown exponentially and is best represented by **Moore's Law** as:

Number of devices in IC = 2 exp [Year - 1960].

Figure 2.1 shows how the device complexity within an IC has increased from 1960 when the first planar silicon transistor was fabricated. This has been largely possible due to the improvement of photolithographic techniques and processing technology. The minimum feature size of the IC has decreased linearly while the size of the chip area has inversely increased with every year [Bar80]. At present about 2 million devices can be integrated employing 2μ technology and this may increase to more than 10

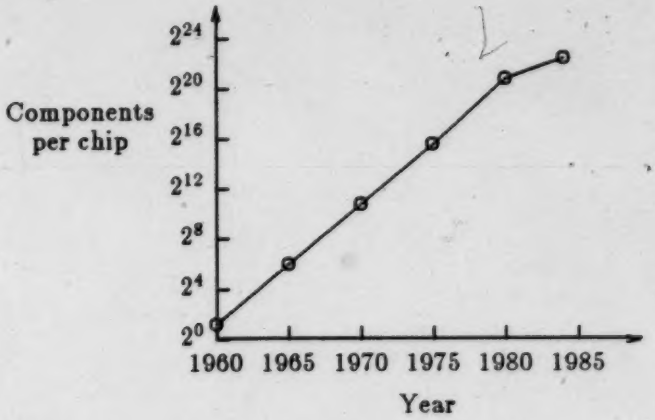


Figure 2.1 Integrated Circuit complexity growth with time

millions by another few years when submicron (0.625μ) technology will be introduced. This will allow the fabrication of several tens of thousands of processors inside an IC and on chip computation employing huge parallelism will be the primary application area of the forthcoming VLSI technology. Thus, the asymptotic analysis similar to program evaluation, is fully justified for the performance analysis of interconnection networks within an IC. The rest of the chapter will propose one such VLSI computational model.

2.3. Rudiments of an Abstract Model

At the highest level of abstraction, a VLSI chip can be thought of consisting of several switches and wires. Switches decide the state of the circuit while the wires are used to connect the switches in a particular configuration. The switches have two mutually exclusive states, called *ON* and *OFF* states. The switches have finite $ON-to-OFF$ and $OFF-to-ON$ transition time. The wires connect the switches and a finite time is required for signal to propagate over the wires. The basic objective

of the computational model is to provide an analytical relationship of these device timings with the physical geometry of the wires and switches. These parameters are highly technology dependent and technological constraints like layout rule, device dissipation, fabricational defects form the part of the model. Metal Oxide Semiconductor (MOS) technology is widely understood to be the best state-of-the-art technology for VLSI fabrication [Gha83,MeC80]. The model discussed in this chapter uses the complementary MOS technology and the criteria for its choice, have been justified in section 2.4.1. In MOS technology, the VLSI circuit consists of three types of wires - metal, polysilicon and diffusion. The wires are embedded vertically in fixed order at different layers - the diffusion at the bottom and the metal at the top. At most, three layers of embedding is possible and this restricts the technology, essentially, to planar embedding. The switches are formed at the cross-points of diffusion and polysilicon wire. Metal can cross-over poly or diffusion wire without making contact. Thus conceptually in a VLSI circuit, switches can be represented as nodes and wires can be represented as edges. This presents an abstraction at the basic transistor level, but the notion can be extended to provide abstraction at higher level of device integration where processors are represented as nodes and interprocessor links are represented as hyper-edges of the graph theoretic model. One such model which will be used to evaluate the performance of interconnection networks is discussed in the next section.

2.4. Computational Model:

The model described here represents the technological and circuit parameters in the form of assumptions and theorems. Each assumption has been justified by discussing the relevant aspects of the state-of-the-art CMOS technology. The assumptions made in this section can be classified into five categories:

2.4.1. Technological Assumptions:
2.4.2. Embedding Assumptions:

2.4.3. Timing Assumptions:
2.4.4. Energy Dissipation Assumptions:
2.4.5. Failure Assumptions:

2.4.1. Technological Assumptions:-

Assumption 1.1: *MOS Technology is most ideally suited for VLSI fabrication.*

The criteria for VLSI technology are high packing density, wide noise margin, good temperature stability, radiation hardness, low fabrication cost, high speed and low power dissipation [Dev80]. Even though {speed}x{power} of MOS technology is lower than that of Bipolar (viz IIL, ECL, STL, etc.), Josephson Junction or Gallium Arsenide technology, the overall performance of MOS technology satisfies the above criteria better than any other technology [Bur83].

Assumption 1.2: *The active devices like transistors, diodes, etc. cannot be embedded vertically one above another.*

At present, the semiconductor technology does not support the concept of three dimensional chip fabrication. Transistors cannot be fabricated vertically one above other from the angle of heat conduction. At present, the third dimension is essentially utilized for heat conduction. Rosenberg [Ros81] has analytically explored the gains and practical difficulties of three dimensional VLSI.

2.4.2. Embedding Assumptions:-

Assumption 2.1: *All processors are identical in shape and size.*

Since the processors have identical computational power, they need equal computational area and hence can be identical in shape and size. Due to the characteristics of the interconnection topology, the optimal drive requirement and fanout capability of the processors may be heterogeneous. To attribute uniformity in size to the processors, either the processors are non-optimally designed so that all the processors have maximum fanout and drive capability or the processors are designed optimally so that

heterogeneous drive requirement can be obtained by constructing drivers on the layout area for wires (i.e., externally to the processors).

Assumption 2.2: *Each processor occupies O(1) area and can be represented as a square of area, A_S.*

The natural layout of any computational circuit has a rectangular topology. In Chapter 4, algorithms are given to transform a rectangular layout of arbitrary aspect ratio (i.e., width to length ratio) into a topologically equivalent square layout. The area expansion due to this transformation has been shown to be bounded by a factor of three.

Assumption 2.3: *Wires have minimal width $\lambda_w \geq \lambda$, where λ is the minimum feature size of the processing technology.*

Usually the value of λ for MOS technology lies between 2μm and 10μm. For Bell Northern Telecom CMOS 1B process $\lambda = 4\mu$m. A lower value of λ helps reduce the computational area and device power consumption. But the undesirable effects like electron-migration and hot spot generation reduces the reliability of the circuit [Bar80].

Assumption 2.4: *At most v_w number of wires are incident on any side of the processor such that $c_1 v_w \lambda_w \leq \sqrt{A_s}$, the factor c_1 is included to take into account the inter-wire spacing.*

For Northern Telecom CMOS 1B process $c_1 = \lambda$.

Assumption 2.5: *Wires always run either in vertical or in horizontal direction in two different layers.*

This scheme is called the Manhattan interconnection technique [Lee61]. Since the processors are aligned parallel to the Cartesian co-ordinates, interconnection wires being perpendicular to the sides of the processors run along the horizontal and vertical

directions.

Assumption 2.6: *At most two wires may cross over each other at any point in the plane.*

In MOS technology metal wires can cross over either polysilicon or diffusion without making contact. Whenever polysilicon runs over diffusion automatic contact is established resulting in a transistor at the overlapping surface.

Assumption 2.7: *Wires can be connected at the cross-over point, if required. This increases the contact resistance and the channel capacitance.*

Assumption 2.8: *The processors are represented as the nodes on planar grid graph and the wires are hyperedges joining two or more nodes.*

Assumption 2.9: *Wires do not cross nodes and also nodes never overlap each other.*

Assumption 2.10: *The minimum physical length of the wire connecting two neighboring nodes is at least* $c_2\lambda_m << \sqrt{A_s}$.

Assumption 2.11 *Wire of length l behaves as a transmission line having* $O(l)$ *resistance and* $O(l)$ *capacitance.*

Since the interconnection wire has distributed resistance and capacitance all along the length of the wire, $O(l)$ assumption is justified.

2.4.3. Timing Assumptions:-

Assumption 3.1: *Wires have unit bandwidth and no broadbanding (i.e., frequency division multiplexing) is allowed.*

Assumption 3.2: *Processors have O(1) computational delay.*

Since the processors are identical in shape and occupies O(1) area, the delay is constant.

Theorem 2.1 *Wire of length, l introduces $O(l^2)$ delay and this is the upper bound of propagation delay.*

Proof: Let R(l) and C(l) be the equivalent resistance and capacitance of the wire of length, l. By assumption 2.11, R(l)=O(l) and C(l)=O(l). From Figure 2.2, the propagation delay, τ_d=ln2.R(l)C(l)=ln2 RC $O(l^2)$=$O(l^2)$, where R and C are the resistance and capacitance per unit square for the wire. For metal wire, R=0.034 Ω/□ and C=2.3 kpF/□. Thus the wire is mostly capacitive and the delay is O(l). For polysilicon wire, R=30 Ω/□ and C=3.2 kpF/□ and the wire is mostly an R-C delay network. This results in a worst case bound of $O(l^2)$ delay. □

Mead and **Conway** [MeC80] have shown a novel way of reducing the delay from $O(l^2)$ to O(l) by interspersing drivers (Figure 2.3) of area O(1) after each k length of wire such that k^2.RC=delay of inverter=O(1). If there are $n=l/k$=O(l) drivers, the total delay of the network will reduce to O(l).

2.4.4. Energy Dissipation Assumption:-

Assumption 4.1 *Each Processor of size O(1) consumes O(1) power.*

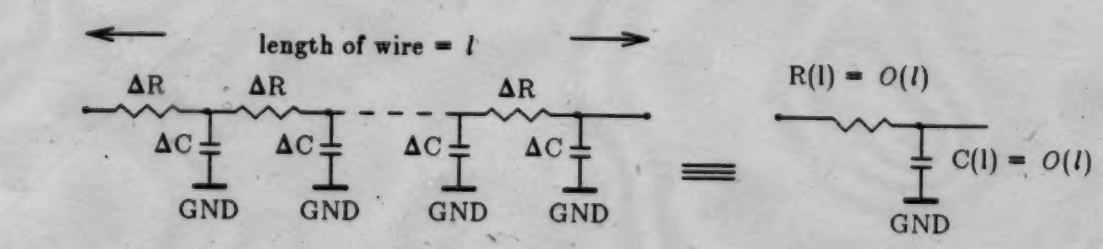


Figure 2.2 Transmission line model of the Interconnection Wire

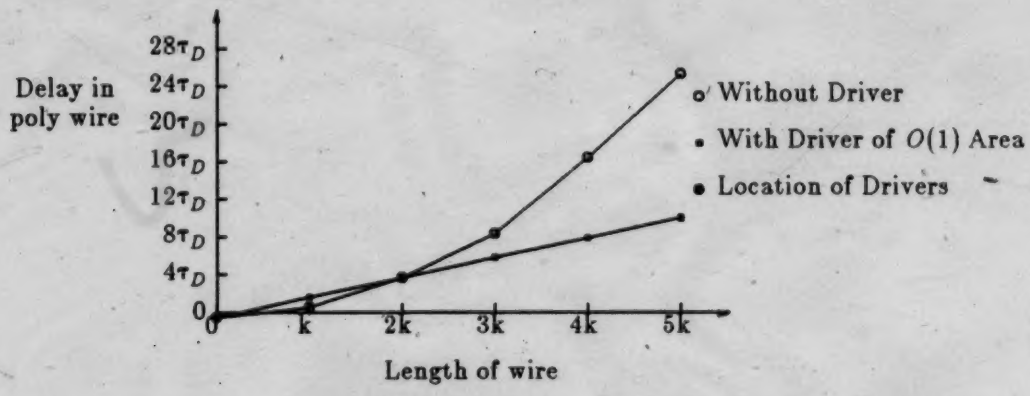


Figure 2.3 Reduction of Delay in Polysilicon Wire by introducing Drivers

Since the power consumption in CMOS occurs due to charging and discharging of nodal capacitors, the average switching power consumption is equal to $0.5CV^2f$, where f is the frequency of operation and $C = O(1)$ is the total capacitance of the processor.

Theorem 2.2: ***Wire of length l consumes at most O(l) power.***

Proof: Refer to Figure 2.2. The power consumed by the resistor $R(l)$ in charging the capacitor $C(l)$ is equal to $0.5C(l)V^2f$. The capacitor stores charge only and does not dissipate any energy. Hence the upper bound is $O(l)$. □

Delay, Area and Power Trade offs:

One of the crucial issues in VLSI interconnection techniques is how to optimize the delay, area and power consumption. As it has been discussed earlier, the delay can be reduced at the cost of area and an increase in chip dissipation. For an interconnection network having long wires, it is desirable to minimize the delay at the cost of increased chip area and dissipation. Above it has been shown how delay can be reduced to $O(l)$ by introducing $O(l/k)$ drivers, each of area O(1). The problem with the **Mead and Conway** technique is that it is difficult to control the location of the

transistors with respect to their delays. A practical method to drive a long wire is to introduce a driver cascade such that the area of the drivers increases monotonically. The propagation delay over the metal wire of length, l can be reduced from $O(l)$ to $O(logl)$ by controlling the area ratio and the number of stages, but this needs an $O(l)$ area for drivers.

Theorem 2.3: ***The metal wire of length, l has a minimum of $O(\log l)$ delay and it needs $O(\log l)$ stages of drivers having a combined area of $O(l)$.***

Proof: Let D_0 be a driver of area 1 and driving resistance R_0 and its input capacitance be C_0, such that it takes $\tau = \ln 2\, R_0 C_0$ time to drive an identical driver. If D_0 drives a wire of length l having C_L capacitance, then it will take $\ln 2\, R_0 C_L = \ln 2 \tau C_L / C_0 = O(l)$ time. If a cascade of drivers consisting of strings $D_1 D_2 \cdots D_{n-1}$ is used (Figure 2.4) such that the area of the i-th driver, D_i is equal to

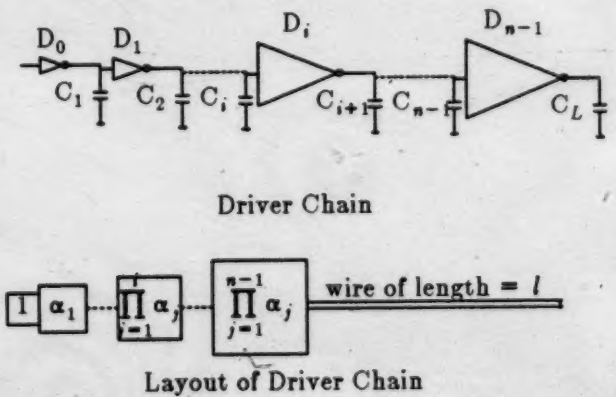


Figure 2.4 Reduction of Delay by introducing a Driver Chain

$\prod_{j=1}^{i} \alpha_j$, then the driving resistance and input and output capacitances of D_i are:

$$1)\ R_i = \frac{R_0}{\prod_{j=1}^{i} \alpha_j}$$

$$2)\ C_i = C_0 \cdot \prod_{j=1}^{i} \alpha_j$$

$$3)\ C_{i+1} = C_0 \cdot \prod_{j=1}^{i+1} \alpha_j$$

Let $\alpha_n = C_L / C_{n-1}$. Then,

$$C_L = C_0 \prod_{i=1}^{n} \alpha_i \tag{2.1}$$

The total delay, T_L is the summation of delays of all drivers and is given by

$$T_L = \tau \sum_{i=1}^{n} \alpha_i \tag{2.2}$$

The total area, A_L occupied by these additional drivers is given by

$$A_L = \sum_{i=1}^{n-1} \prod_{j=1}^{i} \alpha_j \tag{2.3}$$

Equations (2.1), (2.2) and (2.3) can be solved to set up trade offs between the area of the drivers and the total delay. T_L can be minimized by setting $\alpha_{i+1} = \alpha_i = \alpha$ for all i. Applying these values to equations (2.1) and (2.2), the minimum value of propagation delay can be obtained as $T_L = O(n) = O(\log l)$ and the corresponding area is $A_L = \sum_{i=0}^{n-1} \alpha^i = O(\alpha^n) = O(l)$. □

In case there is not enough space to lay down these drivers externally to the processors, it may be necessary to sacrifice the delay for the sake of economy of area. If only $O(l^{1-q})$, where $0 < q < 1$, additional area is available then it can be shown that the penalty in delay is at most $O(l^q)$. Hence,

Theorem 2.4: *A metal wire of length $O(l)$ needs at least $O(l^{1-q})$ driver area to transmit the signal in time $O(l^q)$.*

Proof: Let in Figure 2.4 $\alpha_n = O(l^q)$ be such that $\prod_{i=1}^{n-1} \alpha_i = O(l^{1-q})$ (from 2.1). Now using Theorem 2.3, it can be easily shown that to drive a driver of input capacitance equal to $O(l^{1-q})$ at most a total area of $O(l^{1-q})$ is required employing $(1-q)log\ l$ stages each of $O(1)$ delay. Thus the total delay is no more than $O(l^q)$ iff $\log l < l^q$. □

Ramachandran [Ram82] has shown that if many parallel long wires exist and space available externally to the processors is not sufficient to lay down all the drivers, then the delay can be minimized at the cost of increasing the size of the processors. In the worst case, this results in quadrupling the area of the processor.

2.4.5. Failure Assumptions:-

The failure in VLSI can be classified into three categories:

(1) Chip related,
(2) Packaging related, and
(3) Field operation, as a function of time.

In this thesis, only a specific types of chip related failures will be mentioned, because their occurrences are highly relevant for the evaluation of interconnection networks.

Assumption 5.1: *Defects in the fabrication of an IC (viz., pinhole defects in oxide, defects in photoresist, implant defects, etc.) are randomly distributed and statistically independent.*

Generally, gross imperfections causing large areas of the chip to be bad (i.e., area defects) are detected at slice test and and line defects (like scratches) do not occur in a well-controlled process [Mur64]. The most commonly encountered chip related flaws are random isolated spot defects. Clusters of spot defects also occur, but they can be treated as a single defect since usually the size of the processor is large enough to encompass the whole cluster. Thus the probability of failure of processors is indepen-

dent and the same for all the processors.

The random defects distribution can be assumed to increase linearly with the defect size for small sized defects like pinholes and to decrease as the cube of defect size for large sized defects like defects occurring in diffusion and metal patterns [Sta83]. Using this random defect distribution, it can be shown that the effect of these defects on interconnection is very drastic and the failure rates are related to the aspect ratio of the interconnection wire. Formally,

Theorem 2.6: ***Long wire of length, l and width, λ_w can fail with a probability proportional to*** $O(l/\lambda_w)$.

Proof: Let L be a long wire of length, l and width, λ_w. Let a circular defect of diameter, η occurs randomly on the conducting wire resulting in a hole as shown in Figure 2.5. If the width of the conducting material available for current conduction is sufficient to carry current in normal operation without causing any catastrophic failure, then the pinhole caused by the defect will not have any effect on the wire. Let δ be the minimum width of the wire required for normal operation at a particular current density, then defects of diameter, $\eta < (\lambda_w - \delta)$ will not cause any failure, while

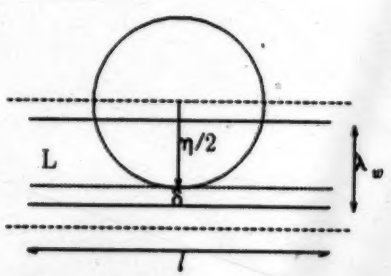


Figure 2.5 Occurrence of Circular Defects of diameter η

the defects of diameter, $\eta \geq (\lambda_w - \delta)$ will cause failures if they occur such that they do not leave δ width of conducting wire. The locus of the center of defects that lead to failure is called *critical area*, $A(\eta)$ and is given by

$$A(\eta) = \begin{cases} 0 & \text{for } 0 \leq \eta < (\lambda_w - \delta), \\ (\eta + 2\delta - \lambda_w)l, & \text{for } (\lambda_w - \delta) \leq \eta < \infty. \end{cases}$$

If $f(\eta)$ denotes the distribution of defect density, then the average value of the critical area with respect to defect size distribution is given by

$$\overline{A} = \int_0^\infty A(\eta) f(\eta) d\eta.$$

Very small defects can be assumed to increase linearly with defect size up to a certain value (say η_0) and large defects can be assumed to vary inversely as the cube of defect size [Sta83]. Thus

$$f(\eta) = \eta/\eta_0^2 \qquad \text{for } 0 \leq \eta < \eta_0,$$

$$f(\eta) = \eta_0^2/\eta^3 \qquad \text{for } \eta_0 \leq \eta \leq \infty.$$

Hence,

$$\overline{A} = \int_{\lambda_w}^\infty (\eta - \lambda_w) l (\eta_0^2/\eta^3)\, d\eta = O(l/\lambda_w), \quad \text{assuming } \delta << \lambda_w.$$

The probability of failure of the wire is directly proportional to its critical area and is $O(l/\lambda_w)$. □

Assumption 5.2: *The yield of an IC reduces with the size of the chip.*

Since the defects occur randomly, the probability of having k defects in a chip is given by **Poisson's** distribution as

$$P(\mu, k) = \frac{\mu^k e^{-\mu}}{k!}$$

where, μ is the average number of defects in a chip and $\mu = A\ D_0$ if D_0 is the average defect density of a chip of size, A. Thus the yield (i.e., probability of having defect free chip) is given by

$$Y_0 = e^{-AD_0} \tag{2.4}$$

Practical results show that this under-estimates the yield slightly for larger size chips [Moo70] and **Poisson's** distribution of random defects represents the yield pessimistically [See67]. A better fitting with practical results can be obtained by employing **Bose-Einstein** statistics and it can be shown [Man81] that the yield of defect-free chips are given by

$$Y_0 = \frac{1}{1+AD_0} \tag{2.5}$$

where, AD_0 is the average number of defects per chip. Figure 2.6 shows yield versus chip area as given by equation (2.4) and (2.5).

Assumption 5.3: *The yield can be improved to a mazimum value by increasing the IC chip area by a constant factor. Further increase in area reduces the yield.*

The yield can be substantially improved by adding redundant processors. The defective cells can be bypassed by employing laser personalization [Elm77] or electrically programmable links [Man81]. **Bose-Einstein** statistics can be applied to analyze

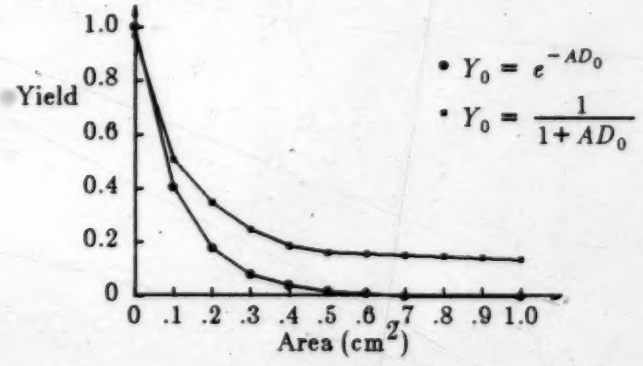


Figure 2.6 Yield vs. Area for Defect Density, $D_0 = 9$ per cm^2

the results of adding redundancy to the chip [Man81]. The result is shown in Figure 2.7 and it is seen that the effect of redundancy is to improve the yield up to certain point (approx. 10-15% of redundancy for well-controlled processes having low defects density) and then the yield decreases with redundancy due to the increased chip area.

2.4.6. Critical Appraisal of the Model:

In the past several researchers suggested various computational models for VLSI. **Brent** and **Kung** [BrK80] and **Vuillemin** [Vui80] have described a computational model where they consider the chip size is very small and propagation delay is constant. Such a model is referred in the literature as a *synchronous model*. Unlike the fixed chip shape (square) described here, **Brent** and **Kung** assume an arbitrary convex planar shape with multi-layers of interconnection. For practical circuits, this generalized model is of little importance, because the chip shape is always square and the interconnection is usually restricted to two layers and they are rectilinear. Additional layers are associated with additional masks and these additional processing complexi-

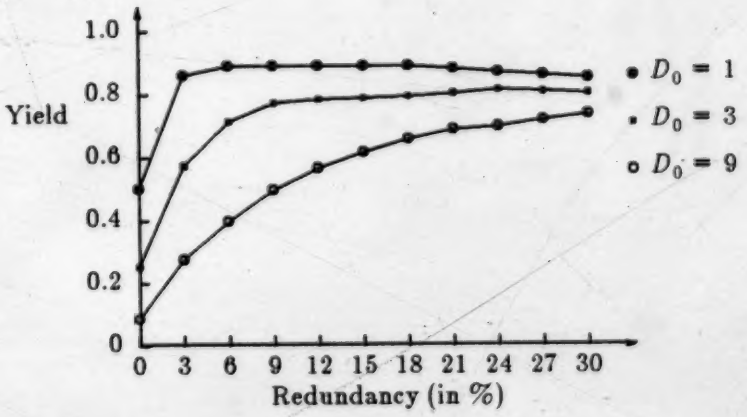


Figure 2.7 Yield vs. Redundancy for Defect Density, D_0

ties reduce the yield considerably. Also, any arbitrary interconnection structure needs at most three layers of interconnection [Pre82]. So, a multilayer interconnection structure like in three-dimensional models [Ros81] is of little practical utility.

Thompson [Tho80], **Ramachandran** [Ram82], etc., have described a computational model where they assume the propagation delay is linear with time. Such models are known as *capacitive models*. If the interconnection consists exclusively of metal wires, such an assumption is valid. Thus the presence of polysilicon wires has been ignored.

Seitz [Sei80], **Chazelle** and **Monier** [ChM81], **Saraswat** [SaM82], **Mead** and **Rem** [MeR82] note that polysilicon wire behaves as a transmission line and the delay is a quadratic function of time. Such models are known as *diffusion models* and **Bilardi,** *et al*, [BPP82] solved the diffusion equation to confirm this behavior of interconnection structure. The upper and lower bounds of propagation delay in the computational model described in this thesis have been computed noting the fact that at present both polysilicon and metal wires are used for interconnection.

Efforts are going on to replace polysilicon by silicides of refractory material like titanium, vanadium, tungsten, etc. These silicides are reported to have ten times lower resistance than polysilicon [Gha83]. Practical problems like adherence of these materials on a silicon surface, fabricational difficulties, etc., restrict their current use in practical circuits. Thus the timing assumptions made in this thesis are quite practical and area-delay trade offs described here determine the practical choice. The earlier computational models are primarily designed to make asymptotic analysis of upper and lower bounds of chip area and computational time. The practical considerations like chip dissipation, processor geometry and, above all, yield versus chip area and interconnection structure have not been addressed. The computational model described in this thesis has been developed to evaluate the interconnection networks from three dimen-

sional viewpoints - area, speed and cost. To date, there is no model in the literature which provides these additional insights. A brief comparison of the proposed model with other computational models is presented in Table 2.1.

2.4.7. Conclusion:

A formal VLSI computational model has been described in this chapter. Relevant aspects of VLSI technology have been formalized by postulating assumptions and theorems under five classes - technological, embedding, timing, energy dissipation and failure occurrences. The model has been compared with other models in the literature and its pragmatic aspects and uniqueness have been identified. Thus the main contribution of this chapter is to propose a formal and practical VLSI model of computation.

Table 2.1 Comparison of Different Computational Models

Type of Assumption	Proposed Model	Thompson's Model	Chaz.& Mon.'s Model	Brent & Kung's Model
On Embedding:				
Processor Geometry:	Square	Rectangular	Convex Polygon	Convex Polygon
No. of Layers:	2	2	≥ 2	≥ 2
On Timing:				
Upper Bound	$O(l^2)$	$O(l)$	$O(l^2)$	$O(1)$
Lower Bound	$O(logl)$	$O(logl)$	$O(logl)$	$O(1)$
On Power Dissipation:				
Wire	$O(l)$	$O(l)$	-	-
Processor	$O(1)$	$O(1)$	-	-
On Failure:				
Defects Density	Random	-	-	-
Chip Yield	$O(1/A_S)$	-	-	-
Wire	$O(l/\lambda_w)$	-	-	-

Chapter 3

EVALUATION OF NETWORKS

3.1. Introduction:

Over the past decade, more than two dozen interconnection networks have been suggested in the literature [AnJ75, Sie79, Thu74]. It is beyond the scope of this thesis to enumerate all of them and evaluate each of them to ascertain its suitability in the VLSI environment. In this chapter, interconnection networks have been grouped into topologically equivalent classes depending on their spatial distribution of the processors and the interprocessor links. One representative from each class has been selected and evaluated employing the computational model described in Chapter 2. Section 3.2 illustrates the rationale behind the classification and the selection of the representative of each class. Section 3.3 enumerates the criteria for evaluation of the interconnection networks under the VLSI environment. Section 3.4 evaluates the two dimensional mesh, the binary tree and the Cube Connected Cycle (CCC). Section 3.5 compares the results of evaluation and section 3.6 concludes that the Cellular Networks like the two dimensional meshes are the most suitable networks for VLSI implementation. This is in direct contrast with the results of **Wittie** [Wit81], **Siegel** [Sie79], etc., who have evaluated the interconnection networks under a non-VLSI environment and concluded that fast networks like the CCC, the PSN, the dual bus hypercubes, etc., are desirable for parallel processing.

3.2. Classification of Networks:

An interconnection network can be represented as a graph $G(V, E)$ where V is the set of processors and E is the set of interprocessor links. The size of the graph is

$N = |V|$, where N is equal to the total number of processors in the network. The edges of the graph are of equal size (say, e) and the total number of the edges, $|E|$ denote the complexity of the graph. The value of $|E|$ is bounded by N and is given by $O(N) \leq |E| \leq O(N^2)$. The class of graphs for which complexity is $O(N^q)$, where $1 < q \leq 2$, are not suitable for VLSI embedding because each node has $O(N^{q-1})$ incident edges and this results in multiple cross-overs. The completely connected networks and the n-dimensional cube are the examples of these graphs. The class of graphs for which the complexity is $O(N)$ have at most $O(1)$ edges associated with each node and will be perused for VLSI embedding. It is interesting to note that for this class of graphs,

$$\lim_{e \to 0} G(V, E) \in \left\{ S_{pl}, S_{tr}, S_{co}, S_{sp} \right\}$$

where, S_{pl} is a planar surface,

S_{tr} is a torroidal surface,

S_{co} is a conical surface,

and S_{sp} is a spherical surface.

The interconnection networks are divided into four topologically equivalent classes depending on whether they describe S_{pl}, S_{tr}, S_{co} or S_{sp}. Cellular Networks like one dimensional array, two dimensional meshes, etc., describe S_{pl}. The networks having a hierarchical connection schema like the binary tree, the X-tree, the hyper tree, etc. describe S_{co}. The networks having wrap-around links like the Plus-Minus 2I wraparound, endaround two dimensional meshes, etc., describe S_{tr}. Networks like the CCC describe S_{sp}. The rationale behind grouping the networks into above four equivalent classes is that for VLSI embedding the edges of the graphs belonging to the same class are stretched similarly. The factor by which an edge is stretched from its original value of e is called the ***dilation***. The maximum value of dilation and the total amount of edge dilation decide the area required by the network for VLSI embedding. Thus, the networks under the same equivalent class asymptotically require similar area

and undergo similar propagation delay.

3.2.1. Identification of networks for evaluation

The strategy adopted in this chapter is to select a representative from each class and investigate its merits for VLSI implementation. The class of networks which describe S_{tr} have wrap-around links and are not suitable for VLSI embedding because of the presence of $O(\sqrt{N})$ number of long wires of length $O(\sqrt{N})$. This results in $O(N)$ delay in the worst case (Theorem 2.1). Also, the presence of these wires poses difficulty in accessing the chip from outside. Therefore, only three networks will be selected from other three classes for detailed study. In this thesis the networks that have been studied are the two dimensional meshes, the binary tree and the CCC. Each of them individually represents its equivalent class. Their selection can be justified as follows:

(1) All the three networks have optimal layout. The two dimensional mesh and the binary tree can be laid in $O(N)$ space which is obviously optimal. The CCC needs $O(N^2/\log^2 N)$ area and this can be shown to be optimal. To permute N numbers, CCC needs $\log N$ steps. Thus, using the $AT^2 \geq O(I^2)$, it is easily seen that the lower bound of area is $O(N^2/\log^2 N)$.

(2) Algorithmic capability of all the three networks have been extensively studied in the literature [Ata83, PrV79].

(3) Practical machines like ILLIAC IV, SOLOMON, X-Tree Machines, etc., have been built employing these topologies.

3.3. Criteria for Evaluation:

The interconnection networks have been modeled to study three aspects - a) the physical aspects, b) the computational aspects and c) the cost aspects. The overall performance of the networks has been estimated from these viewpoints which form three

orthogonal classes.

3.3.1. Physical aspects:

The physical aspects relate to the chip area and power consumption by the chip.

3.3.1.1. Chip Area

The chip area refers to the total area required to lay the processors and the communication links. The area occupied by the processors is the computation area. The ratio of the computation area to the total area is the performance metric and is defined as *area efficiency*.

3.3.1.2. Power Consumption

Like the chip area, the total power consumed by the chip can be divided into computational power which is equal to the power dissipated by the processors and the power required to drive the communication links. The ratio of the computational power to the total power consumed by the chip is defined as the *power efficiency*.

3.3.2. Computational Aspects:

The computational aspects refer to the speed of computation and the message flow within the networks. The speed is estimated by computing the delay in interprocessor communications and the message density in the interprocessor links reveals the message flow.

3.3.2.1. Delay

The delay refers to the time needed in interprocessor communication to execute a computational task. This is both the property of the topology of the network and the length of interconnects. The propagation of signals in VLSI has been analyzed in section 2.3. The network topology will be analyzed and the results of the computational

model in Chapter 2 will be employed to compute the delay of the networks. The average delay and the worst case delay are the measures of computational speed of the networks.

3.3.2.2. Message Traffic Density

An important aspect of the computational power of the networks is the distribution of data flow within the networks. An efficient network should avoid the message traffic congestions at the links and should distribute the data (message) flow uniformly across all the available links. The message traffic density at the links with respect to the networks size is a good measure of the computational power of the network. In the analyses made in this Chapter, the average rate at which each node originates messages is assumed to be fixed at one message per time unit regardless of network size N. Also it has been assumed that the average rate at which any node, i within the network transmits messages to another node $j \neq i$ in the network, is constant.

3.3.3. Cost Aspects:

The cost aspects consider the fabrication cost and the replacement cost due to poor reliability of the networks. The manufacturing cost of the IC is related to the total chip area (Assumption 5.2) and the regularity of the layout [Seq83]. The reliability of the networks largely depends on the presence of long interconnects (Theorem 2.6) in the embedding and the topology of the networks. Depending on the existence of alternate message routes within the networks, the networks can fail completely or partially.

3.3.3.1. Yield

The yield of the IC is largely dependent on the total chip area. The occurrence of random spot defects will reduce the yield by a factor inversely proportional to the area, A of the chip. This $O(1/A)$ factor is called the *yield factor* and is a measure of

manufacturing cost of the IC. The defective processors can be replaced by redundant processors, but the chips with defects on the interconnects cannot be salvaged. The size of the interconnect is highly relevant for the chip yield (Theorem 2.6) and the presence of long wires will be enumerated for evaluation of the manufacturing cost.

3.3.3.2. Regularity

The regularity of the network largely decides the layout cost. Since all processors are identical, an $O(1)$ layout cost can be assumed for laying out a processor. Each link between the processors also can be made hierarchically, the actual cost to layout the links may cost much less than the actual number of links. The *regularity factor* is a measure of layout cost and is defined as the ratio of total number of interconnections to the number of interconnections actually laid.

3.3.3.3. Fault-tolerance

The fault-tolerance capability largely decides the reliability of a working chip. Due to a host of causes, like electromigration, Kirkendall's effects, hot electron effects, etc., a processor may fail during the normal use of the chip. Depending on the topology of the networks, the effect of failure of a single processor will adversely affect the operation of the network. The network reliability can be graded depending on the effects of the failed processor on the performance of the network. If the failed processor is disabled and the computation by the rest of the processors can be done without the need of any additional component, then the fault-tolerance capability can be considered as **High.** If the failed processor eliminates a group of processors from the network or causes the entire network to fail, then the fault-tolerance capability can be considered **Low** and the redundant components are necessary to be introduced within the chip.

3.4. Evaluation of Networks:

3.4.1. Two Dimensional Meshes

The two dimensional mesh is shown in Figure 3.1 and the VLSI layout for the networks is shown in Figure 3.2. By assumption 2.2, each processor is represented as a unit square and by assumption 2.10, the interconnect length can be ignored. Thus both the area efficiency and the power efficiency of two dimensional meshes are approximately equal to 1.

In order to compute the delay in a $\sqrt{N} \times \sqrt{N}$ mesh, it should be noted that the message path length between two arbitrarily located processors at (i, j) and (k, l) within the square grids is

$$d = \sqrt{(i-k)^2 + (j-l)^2}$$

Clearly, $d_{max} = 2\sqrt{N}$ corresponding to the processors at the end of a diagonal and $d_{min} = 1$ corresponding to two neighboring processors. Assuming an $O(1)$ delay time

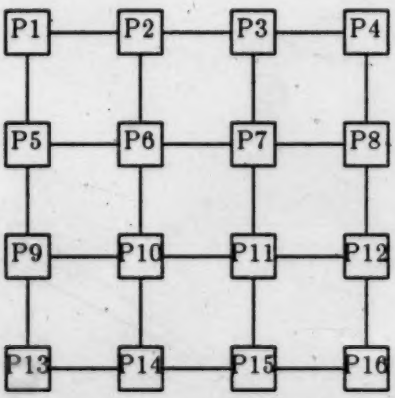


Figure 3.1 A 4 x 4 Mesh Network

(Assumption 3.2) associated with each processor, the worst case delay is $O(d_{max}) = O(\sqrt{N})$. The average delay can be calculated from the average path length. The average message path length in the square grids is equal to $(1+2+\cdots+2\sqrt{N})/2\sqrt{N} = \sqrt{N}+0.5$ and hence the average delay is also $\overline{D} = O(\sqrt{N})$.

The total number of links in the square mesh is equal to $2\sqrt{N}(\sqrt{N}-1) = O(N)$ and the average message path is $O(\sqrt{N})$. Assuming all the N nodes issue messages simultaneously, the average message traffic density is then $\overline{M} = NO(\sqrt{N})/O(N) = O(\sqrt{N})$.

Since the area efficiency is 1, the chip size is small and by assumption 5.2, the yield is $O(1/N)$.

The layout can be constructed hierarchically and a block of 4^k processors can be laid in $k-$th step paying 2^{k+1} cost. Thus a network of size N needs $\sum_{k=1}^{\log_4 N} 2^{k+1} = 2^{\log_4 N+2}-4 = O(\sqrt{N})$ cost. The regularity factor is thus $O(N)/O(\sqrt{N}) = O(\sqrt{N})$.

P1	P2	P3	P4
P5	P6	P7	P8
P9	P10	P11	P12
P13	P14	P15	P16

Figure 3.2 Layout of 4 x 4 Mesh Network

The failure of single processor does not impair the performance of the networks drastically. Due to the presence of parallel paths in the square grids, the effect of a failure of a single processor can be classified as **Low.**

3.4.1.1. Discussion of Related Networks:

The delay and message density can be improved for mesh networks if the processors belonging to each column and each row are connected hierarchically as binary trees (Figure 3.3). Such networks are known in the literature as orthogonal tree networks [NMB83], mesh of trees [Lei82, Ull84] and orthogonal forests [CaS81]. The average delay for such networks reduces to $O(logN)$ but the chip area increases to $O(Nlog^2N)$. The overall performance thus does not improve. On the contrary, the presence of long interconnects of length $O(\sqrt{N})$ actually increases the average delay to $O(log^2N)$ (by Theorem 2.3). Moreover, these networks suffer from many practical limitations like poor yield (due to large chip size), poor regularity (due to presence of mesh and trees combined), $O(N\log N)$ cross-overs, long interconnects, etc.

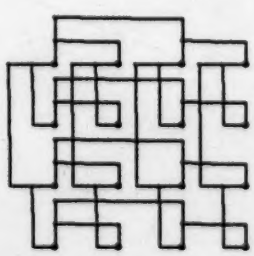


Figure 3.3 Mesh Network with Tree Interconnection

3.4.2. Binary Tree Networks:

The complete binary tree networks of N processors is shown in Figure 3.4 which needs at most $O(N\log N)$ layout area corresponding to $O(N)$ leaves and $O(logN)$ height of the tree. A better layout which needs optimal $O(N)$ area can be constructed using the concept of an H-diagram, originally proposed by **Marihugh** and **Anderson** [MaA71] as a graphical approach to logic design. **Horowitz** and **Zorat** [HoZ81] have constructed the algorithms for the generation of such a layout and the modified network is henceforth referred as an **H-tree**. The H-tree network is shown in Figure 3.5 and the corresponding layout is shown in Figure 3.6. The total area for a complete binary tree of N processors can be computed from the following recursive relationship

$$A(N) = [2\sqrt{A(\lceil N/4 \rceil)} + 1]^2 \qquad \text{with} \quad A(1) = 1.$$

If $S(N) = \sqrt{A(N)}$ is the side of the square layout, then

$$S(N) = 2S(\lceil N/4 \rceil) + 1 \qquad \text{with} \quad S(1) = 1.$$

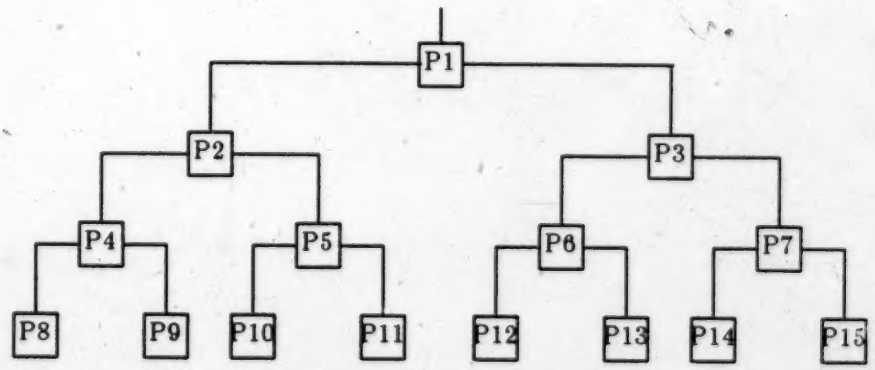


Figure 3.4 Layout of a Binary Tree

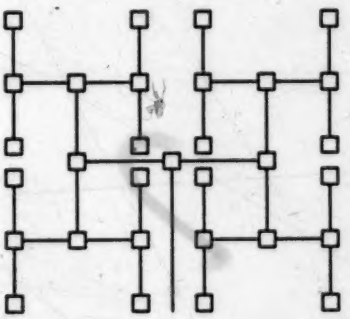


Figure 3.5 The H-Tree Network

Assuming $N = 2.4^k - 1$, $S(N) = \sum_{j=0}^{k} 2^j = 2^{k+1} - 1$.

Therefore, $A(N) = 4^{k+1} - 2^{k+2} + 1 = 2N - 2.82\sqrt{N+1} + 3$.

Thus, the area efficiency is better than 0.5. The longest wire in the layout is of size $\sqrt{N}/2$ and the total length of wires in the layout is given by the recurrence relation:

$$L(N) = 4L(\lceil N/4 \rceil) + \sqrt{N} \qquad \text{with} \quad L(7) = 1$$

which gives a solution $L(N) = O(\sqrt{N}\log N)$.

The power efficiency depends on $L(N)$ and the width of the wires and is more than 0.5.

The worst case delay occurs when a message is propagated from the root to any of the leaves. Assuming $O(l)$ driver space is available with each wire of length l, the worst case delay can be given by $D_{max}(N) = O(log\sqrt{N}) + O(log\sqrt{(N/4)}) + \cdots = O(log^2 N)$ (by Theorem 2.3).

To calculate the message traffic density on a link, consider $N-1$ time units during which $N(N-1)$ messages are generated and each node on the average will have

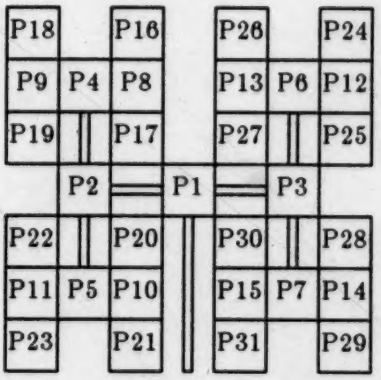


Figure 3.6 Layout for H-Tree Networks

sent a message to each of the others. Let $h = \log(N+1)$ be the height of the tree network, denoted as T_h, such that $|T_h| = N$. A subtree of T_h at level k from leaves is shown as T_k such that $|T_k| = 2^{k+1} - 1$. A link between level k and level $k+1 \leq h$ will be used to transmit messages between i) all the nodes of the left and the right subtrees each of size $|T_k|$ and ii) one subtree of size T_k connected by the link and $(N - 2T_k)$ nodes of the tree, T_h (Figure 3.7). Thus, the message density per unit time at a link between level k and level $k+1$ is

$$M(k) = \frac{2}{N-1}[|T_k|^2 + (N - 2|T_k|).|T_k|)] \approx 2(2^k - 1)\left(1 - \frac{2^k - 1}{2^{h+1} - 2}\right)$$

Clearly, at links above the root, $M(h+1) = 0$ and at links connecting the leaf nodes $M(1) = 2 - 2/(2^{h+1}-2) \approx 2$. The maximum congestion occurs at the link $k = h$ for which $\frac{\partial M(k)}{\partial k} = 0$. Thus for a binary tree, the message traffic density is highest in the links connecting the root and its sons. The total number of messages that pass through the root is $M(h) = 2(2^h - 1)\left(1 - \frac{2^h - 1}{2^{h+1} - 2}\right) = 2^h - 1 \approx \frac{N}{2} = O(N)$.

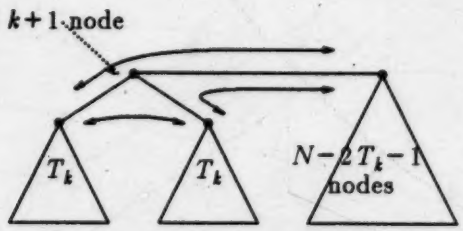


Figure 3.7 Message Flow through a level k node in a Tree Network

Since the area efficiency is less than 1, the yield is not likely as high as meshes, but it is asymptotically equal to $O(1/N)$.

The layout can be constructed hierarchically and each level of embedding needs $7\times O(1)$ cost and the overall connection cost for a network of N processors is equal to $O(\log_4 N)$. Thus the regularity factor is $O(N)/O(logN) = O(N/logN)$.

The fault-tolerance capability of the network due to the failure of a single processor depends on the location of the processor. If the external communication is done solely through the root, its failure will have total disastrous effect invalidating the usability of the IC. Since there is no parallel path for message flow, failure of any links will truncate the operability of the chip. If any processor other than the root fails, will also reduce the performance of the IC by an amount depending on its location in the tree. The network can be restored to function normally by replacing the defective processor by redundant processors. Redundant processors can be placed in the extra space available within the chip and re-routing can be done by electrically programmable routing or laser personalization.

3.4.2.1. Discussion of Related Networks:

The fault-tolerance capability of the network can be improved by putting additional links as shown in Figure 3.8, the modified tree network is referred in the literature as an X-tree [DeP79]. Since the degree of nodes in the tree is never more than 5, the overall chip area is $O(N)$. But the layout involves many cross-overs and long outer edges. A better layout without many crossovers can be obtained by connecting the leaf nodes only so that each node has degree 3. But the network has reduced fault-tolerance capabilities, even though the worst case delay improves. The increase of chip area by several orders in these networks reduces the efficiencies and yield of the IC and the gain of fault-tolerance is off-set. A better tree network which has the same chip area like the H-tree, but has improved capability as far as the delay and the fault-tolerance are concerned, is shown in Figure 3.9.

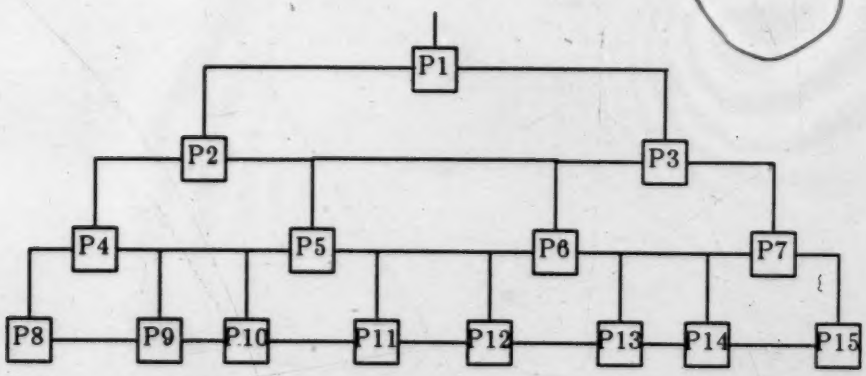


Figure 3.8 Layout of an X-Tree

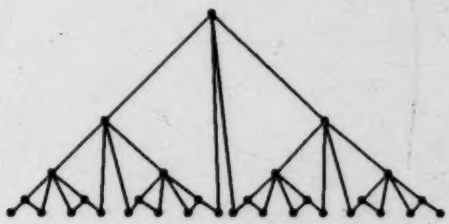


Figure 3.9 Modified Binary Tree suitable for VLSI Implementation

3.4.3. Cube Connected Cycles:

An m dimensional Cube Connected Cycle (CCC) is a network which can be derived from a boolean hypercube of 2^m vertices by replacing each vertex with a cycle

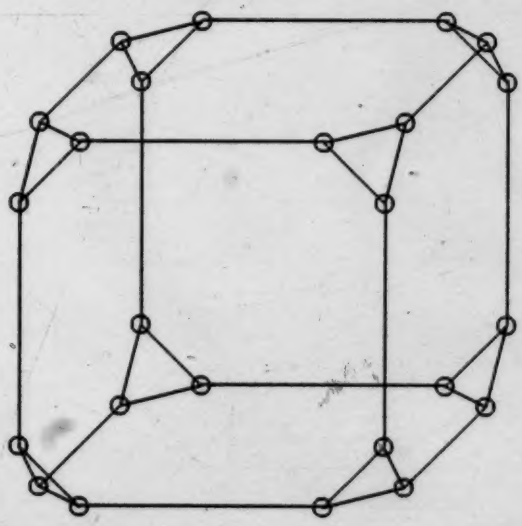


Figure 3.10 Cube Connected Cycle topology for 3. 2^3 processors

of m vertices. This guarantees that the order of each vertex is of degree 3 and not $O(logN)$ as in a boolean hypercube. The topology of a 3-dimensional CCC with $3.2^3 = 24$ processors is shown in Figure 3.10 and its optimal VLSI layout, originally due to **Preparata** and **Vuillemin** [PrV79], has been given in Figure 3.11. In order to justify that the layout is optimal by **Thompson's** [Tho80] lower bounds on AT^2, it should be noted from Figure 3.11 that a 3-dimensional CCC needs $2^3 \times (2.2^3 - 1)$ layout area and by induction an m dimensional CCC needs $2^m \times (2.2^m - 1)$ chip area. Since $N = m2^m$, then $2^m = N/(logN - \log m)$, i.e., $m \approx \log(N/logN)$. Thus the total chip area is approximately $(N/logN) \times (2N/logN - 1) = O(N^2/\log N^2)$. Thus both the area efficiency and the power efficiency are $O((\log^2 N)/N)$.

Since the total area required for the IC is $O(N^2/\log^2 N)$ and the area occupied by the processors is equal to $O(N)$, the average wire length is $(N/\log^2 N)$. Wires are either horizontal or vertical and cannot be both metal (Assumption 2.5). Thus the average delay is $O(N/\log^2 N)$, by introducing interspersed drivers of area $O(1)$ (Figure 2.5). The

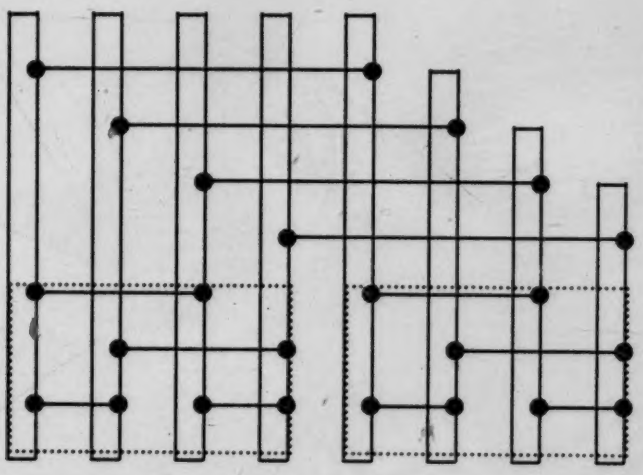


Figure 3.11 Layout of CCC with 3.2^3 processors

worst case delay is due to message transmission between two processors at the opposite edges of the chip and is proportional to the perimeter of the chip, i.e. $O(N/\log N)$. Since each node has degree 3, the total number of links is equal to $3N$. If N messages are generated on the average in unit time, then the average message density is equal to $\overline{M} = (N/3N) \times O(N/\log^2 N) = O(N/\log^2 N)$.

Since the chip area is very large, the yield is $O(\log^2 N/N^2)$, which is very poor compared to the mesh and the tree networks.

The layout can be partially constructed hierarchically. It needs $O(N)$ connections as is evident from Figure 3.11. Since the total number of connections in the layout is equal to $3N$, the regularity factor is $O(1)$ and is extremely poor compared to the mesh and the tree networks.

The fault tolerance capability of CCC is good because there is always an alternate path to re-route the message like in the mesh. Thus the failure of a single processor will not have any drastic effects on the performance of the network.

3.4.3.1. Discussion of Related Networks

One alternative network to CCC which has also an optimal layout area of $O(N^2/\log^2 N)$ is the Perfect Shuffle Network (PSN). The best practical layout of PSN known to date is due to **Rodeh** and **Steinberg** [StR80] and needs $O(N^2/\log^{3/2} N)$ area. Thus, area efficiency, power efficiency, average delay, mesage density, yield, etc., are slightly poorer than CCC. Even if the optimal practical area is obtained, these quantities will never be asymptotically better than CCC. Moreover, the less regularity in a PSN layout is a genuine disadvantage. Networks like Butterfly, Hypercube, etc., have long interconnects like CCC and share similar disadvantages.

3.5. Comparison of Three Classes of Networks:

From the analyses done in the previous section, it is evident that each network has certain strong aspects and certain weak aspects. It is difficult to relate all these aspects by a compact formula which can be utilized as a performance metric. A weak effort in this respect was originally done by **Mead** and **Rem** [MeR82] relating the area and the speed and proposing area×speed as a metric. **Thompson** has extended the concept for any arbitrary network by showing that area ×(speed)^2 indicates a better performance metric. **Savage** has contended that (area)^2×speed reflects a better evaluation for certain computational problems like binary sorting. From all these contradictory claims, it is evident that it is virtually not possible to correlate all the criteria discussed in section 3.5. An alternative strategy like **Wittie's** [Wit81] order of five magnitude evaluation technique has been employed here. The networks have been given credit points for each criterion and the total points have been used as a performance index for the network. The results of an evaluation with respect to different criteria have been shown in Table 3.1. The credit points have been assigned on the basis of relative merits of the networks. From the values of total points, it can be seen that the two dimensional mesh networks indicate overall better performance than the tree and the CCC. This is in direct contrast to the results of **Wittie** [Wit81], **Siegel** [Sie79], etc., who have concluded that fast networks like CCC, PSN, spanning bus hypercubes, etc., have better overall performance.

3.6. Conclusion:

The basic conclusion from this chapter is that the Cellular Networks which have a similar structure to the mesh can be cost effectively implemented for VLSI implementation and are highly suitable for VLSI parallel processing. The penalty in delay can be offset by the gains of several criteria discussed in this chapter. The main contribution of this chapter is to propose a practical set of criteria for the evaluation of

Table 3.1 Evaluation of Three Classes of Networks

Evaluation of Meshes, H-Tree and CCC for VLSI application with respect to Physical, Computational and Cost Aspects								
NETWORK STRUCTURE	EVALUATION CRITERIA AND PERFORMANCE							OVERALL RESULTS
	Average Message Delay	Average Message Density	Area Efficiency	Power Efficiency	Chip Yield Factor	Layout Regularity Factor	Fault Toler -ance	
MESHES	$O(\sqrt{N})$	$O(\sqrt{N})$	1	1	$O\left(\frac{1}{N}\right)$=Max	$O(\sqrt{N})$	High	
	2	3	3	3	3	2	3	19
H-TREE	$O(\log^2 N)$	$O(N)$	<1	<1	$O\left(\frac{1}{N}\right)$<Max	$O(\log N)$	Low	
	3	1	2	2	2	3	1	14
CCC	$O\left(\frac{N}{\log^2 N}\right)$	$O\left(\frac{N}{\log^2 N}\right)$	$O\left(\frac{\log^2 N}{N}\right)$	$O\left(\frac{\log^2 N}{N}\right)$	$O\left(\frac{\log^2 N}{N}\right)$	$O(N)$	High	
	1	2	1	1	1	1	3	10

networks and to employ them on three classes of networks. A conceptual classification on the basis of spatial topology readily justifies the above conclusion from the results of comparison in section 3.5.

Chapter 4

ALGORITHMS FOR LAYOUT TRANSFORMATION

4.1. Introduction:

In order to minimize the delay both along the horizontal and the vertical directions of the chip, it is recommended that each processor should be equal in size and square in shape [Lei81]. Therefore, in the VLSI computational model of Chapter 2, the processors have been assumed to have square shape (Assumption 2.2). *But the cruz of the problem is how to transform a natural layout (having a rectangular shape) into a square layout.* In section 4.2 of this chapter, three simple algorithms have been designed to transform a rectangular layout of arbitrary aspect ratio into a square layout whose area is asymptotically linear to the area of the rectangular layout. The limitations of such techniques have been discussed in section 4.3. Finally in section 4.4, alternative viable techniques are suggested instead of postprocessing an optimal rectangular layout into a square layout.

4.2. Algorithms for layout transformation:

The most obvious way to develop a layout transformation is to cut the rectangular layout into suitable blocks and place them within a square area such that the blocks are interconnected to restore the circuit topology of the original layout. In order to facilitate the design of simple and efficient embedding algorithms, the blocks are assumed to be rectangular or square in shape. Thus the blocks can be made by employing either simple vertical cuts across the width of the layout or at suitable distances both along the width and the length. Thus the embedding techniques due to bidirectional cutting fall under the class of problems of embedding rectangular grids

into square grids [AlR80].

4.2.1. Break and Fold Embedding:

The easiest technique of layout transformation is by using the notion of cutting the layout along its length into suitable size blocks and placing them within a square area as shown in Figure 4.1. Algorithm 4.1 computes the size of the square area and the location of cuts from the input parameters of length and width of the rectangular layout. Since the layout is broken and folded like a carpenter's ruler, it is called the **Break and Fold Embedding.**

The factor $e(\epsilon)$, in Algorithm 4.1, is defined as the expansion of embedding, ϵ and is defined as the ratio A_S/A_R, where A_S is the area of the square layout and $A_R = LW$ is the area of the rectangular layout.

If (i, j) and (k, l) are the cartesian co-ordinates of two points in the rectangular layout such that $\sqrt{(i-k)^2 + (j-l)^2} = 1$ and $\epsilon(i, j)$ and $\epsilon(k, l)$ are the co-ordinates of the corresponding points in the square layout, then the distance, $\delta(\epsilon) \geq 1$, between

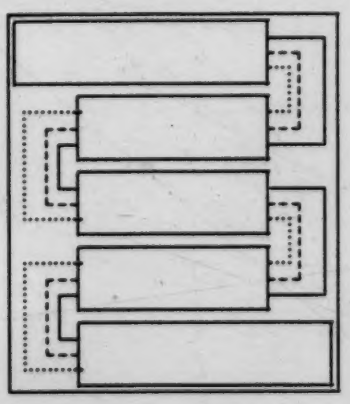


Figure 4.1 Transformation of Layout by Break and Fold

Algorithm 4.1: Break and Fold

Input: L, W /* L and W are length and width of the rectangular layout */

Begin

$\sigma := \frac{W}{L}$;

(1) **If** $\sigma > \frac{1}{3}$ **Then** Embed the layout as it is within the square of side L;

Else,

(2) **Begin**

$$k := 2 + \frac{\frac{1}{\sigma} - 2}{\left\lceil 1 + \left(\frac{1}{\sigma} - 1\right)^{\frac{1}{2}} \right\rceil}$$

$e(\epsilon) := \sigma k^2$

$s := kW$.

(3) **For** $j := 1$ **Step** 1 **Until** k **Do**

Begin

Break the layout at the length $js - (2j-1)W$ from the left;
Embed the pieces as shown in Figure 4.1;
Reconnect the severed wires as shown in Figure 4.1;

End {For};

End {Else};

End.

$\epsilon(i, j)$

and $\epsilon(k, l)$ is the stretching due to embedding. The factor $d(\epsilon) = \max \delta(\epsilon)$ for all pair of points in the layout is called the **Dilation of Embedding**, ϵ.

Theorem 4.1: *The upper bound of expansion by Algorithm 4.1 is 3.*

Proof: [Lei81]

For $\sigma > 1/3$, by line 1 of Algorithm 4.1, the area of the square is $A_S = L^2 < 3A_R$ such that $c(\epsilon) > 3$.

To prove the theorem for $\sigma \leq 1/3$, it is sufficient to prove that Algorithm 4.1 can fold in a rectangle of width W and length $l \geq L$ into a square of area $3LW$. Since $\sigma \geq 1/3$, by Algorithm 4.1, $s \geq 3W$ such that by line 3, there will be at most $(k-2)$ pieces of length $s - 2W \geq s/3$ and 2 pieces of length $s - W \geq 2s/3$ such that the lower bound of the length of the maximum sized rectangle is $l \geq (k-2)s/3 + 2(2s/3) = s(k+2)/3$.

Since $k = \left\lfloor \sqrt{3/\sigma} \right\rfloor$ is the largest number of pieces of width W that can be folded into the square, it follows that $k+1$ pieces of width W will not fit. Therefore, the length s of the side of the square must be strictly less than $W(k+1)$, which means

$$s \leq W(k+1) - 1.$$

Also, $(s+1)^2$ must be strictly larger than $3A_R$, and hence

$$3LW \leq (s+1)^2 - 1 = s(s+2).$$

Substituting for s,

$$3LW \leq s(W(k+1) - 1 + 2) = sW(k+2)$$

Since $W \geq 1$, dividing both sides by $3W$, yields $L \leq s(k+2)/3$. But the right hand side of this inequality is equal to the lower bound of the maximum sized rectangle that can be folded into the square of area A_S. □

Even though the expansion factor is linearly bounded, the main disadvantage of the above embedding scheme is that a skewed delay is introduced along the lateral direction of the layout. Since the dilation of embedding is equal to $4W$ (Figure 4.1), this delay is bounded by the width, W, of the layout and in the worst case can be

$O(W^2)$ for polysilicon interconnection (by Assumption 2.11).

In order to reduce the dilation, it is imperative to introduce bidirectional cutting of the layout across its length and width. Let the rectangular layout of length $L=\alpha n$ and width $W=\alpha m$ be sliced into mn blocks such that there are $n-1$ slices parallel to its width and $m-1$ slices parallel to its length. Thus the original layout can be represented as a grid graph $G(m, n)$ of m rows and n columns. The nodes of the graph are the blocks each of size, α^2 and the edges of the graph represent the adjacency of the nodes. Thus the layout transformation problem is essentially embedding a rectangular grid graph, $G(m, n)$ into a square grid graph, $G(s, s)$. The doublet (i, j) refers to the block located at the intersection of i-th row and j-th column. Two simple algorithms for such embedding are discussed here.

4.2.2. Step Embedding:

The embedding scheme is pictorially represented in Figure 4.2. Under the embedding scheme, row i of $G(m, n)$ is identical to row i of $G(s, s)$ until the location

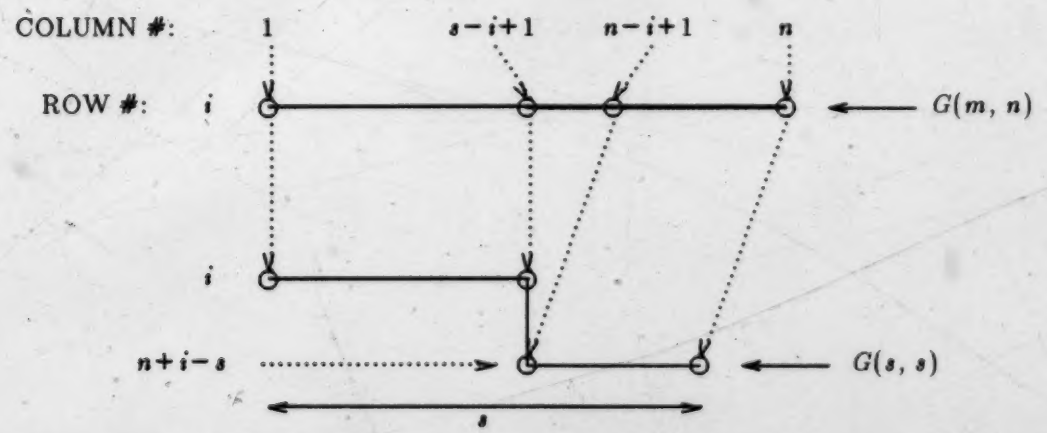


Figure 4.2 Layout Transformation by Step Bending

$<i, s-i+1>$;

it then proceeds vertically along column $s-i+1$ of $G(s, s)$ until the location $<i, s-i+1>$; it then proceeds vertically along column $s-i+1$ of $G(s, s)$ until the location $<i+n-s, s-i+1>$, and it then proceeds rightward along row $i+n-s$ of $G(s, s)$ until column n is exhausted. Algorithm 4.2 determines the co-ordinates of the blocks in the rectangular layout on the square grids. The (i, j)-doublet, i.e., the block corresponding to (i, j)-th node in $G(m, n)$ is denoted as $\epsilon(i, j)$-doublet in $G(s, s)$ graph and the embedding algorithm provides the positional co-ordinates of $\epsilon(i, j)$ on $G(s, s)$. The algorithm is called **Step Embedding** because the blocks are rearranged to represent a Step Function (as in Figure 4.2).

From Figure 4.2, it is evident that $d(\epsilon) = 3$ and $e(\epsilon) = s^2/mn$. The expansion factor is small when $n = m$, and increases monotonically with $\sigma = m/n$.

4.2.3. Embedding by Folding

This algorithm provides a better bound of expansion for any value of σ. The embedding scheme is pictorially represented in Figure 4.3. Under the embedding, row i of $G(m, n)$ is identical to row i of $G(s, s)$ until the location $<i, s-i+1>$. It then proceeds down diagonally at an angle of $45°$ until location $<2i-1, s>$ and then goes vertically down to $<2i, s>$. Then it proceeds towards left at first at an angle of $45°$ until $<m+i, m-i+1>$ and then horizontally till $<m+i, s-i+1>$ and finally once again at an angle of $45°$ till $<m+2i-1, 1>$.

Algorithm 4.3 determines the co-ordinates of $\epsilon(i, j)$ on $G(s, s)$. Since the algorithm folds the layout like Algorithm 4.1, it is called **Embedding by Folding**.

The dilation due to the Algorithm 4.3 is equal to 2 and the expansion is equal to $\left\lceil \sqrt{n/m} \right\rceil^2 \times (m/n)$.

Algorithm 4.2: Step Embedding

Input: m, n

Output: Co-ordinates on $G(s, s)$ for all doublets of $G(m, n)$.

Begin

$s := \left\lceil \frac{m + n}{2} \right\rceil$;

$i := 1$;

(1) **While** $i \leq m$ **Do**

Begin

(2) **For** $j := 1$ **Step** 1 **Until** $s-i+1$ **Do**
$\epsilon(i, j) := \langle i, j \rangle$;

(3) **For** $j := s-i+2$ **Step** 1 **Until** $n-i+1$ **Do**
$\epsilon(i, j) := \langle 2i+j-s-1, s-i+1 \rangle$;

(4) **For** $j := n-i+2$ **Step** 1 **Until** n **Do**
$\epsilon(i, j) := \langle i+n-s, j+s-n \rangle$;

$i := i+1$;

End {while}

End.

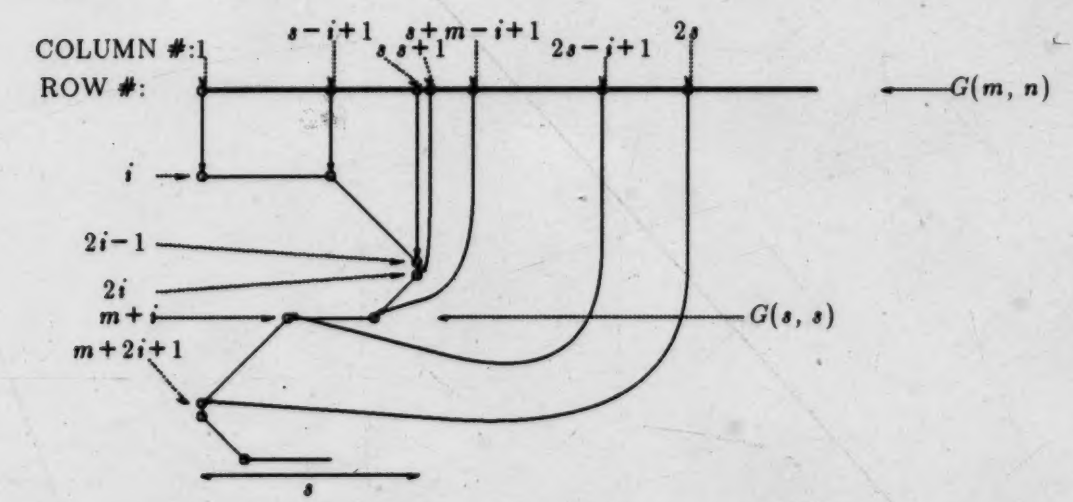


Figure 4.3 Embedding by Folding

Algorithm 4.3: Embedding by Folding

Input: m, n ;
Output: Co-ordinates on $G(s, s)$ for all doublets of $G(m, n)$.

Begin

$s := m \left\lceil \left(\frac{n}{m} \right)^{\frac{1}{2}} \right\rceil$;

$q := \left\lfloor \frac{n-1}{s} \right\rfloor$;

$i := 1$;

(1) **While** $i \leq m$ **Do**

Begin

(2) **While** $j \leq n$ **Do**

Begin

$Q := \left\lfloor \frac{j-1}{s} \right\rfloor$;

$R := j - sQ$;

$l := R\,[(Q+1) \bmod 2] + (m+1-R)(Q \bmod 2)$;

(3) **If** $[(Q=0)\ AND\ (R \leq s-i+1)]\ OR$
$[(R \geq m-i+1)\ AND\ (R \leq s-i+1)]\ OR$
$[(Q=q)\ AND\ (R \geq m-i+1)]$
Then $\epsilon(i, j) := <i+mQ, l>$;

(4) **Else**

Begin

If $[(Q<q)\ AND\ (R \geq s-i+1)]$ **Then**
$\epsilon(i, j) := <2i+mQ+R-s-1, l>$;

(5) **Else** $\epsilon(i, j) := <2i+mQ+R-m-1, l>$;

End {If} ;

$j := j+1$;

End {While j} ;

$i := i+1$;

End {while i} ;

End.

4.3. Discussion of results:

From the above results, it is evident that a rectangular layout can be, in principle, transformed into a topologically equivalent square layout. The expansion and dilation due to transformation are small if the layout is sliced both vertically and horizontally. But, for practical circuits, which is optimally laid on the rectangle, this bidirectional slicing may lead to deleterious effects. Also, the algorithms 4.2 and 4.3 assume that the total width of the interconnection is much smaller than α. In practice, an additional $O(s)$ area is required to accommodate the rectilinear interconnections to run around the nodes. But, the biggest shortcoming of such embedding techniques is that they tend to shift the external pins located on the edges of the rectangle into the interior of the square. Like the H-tree layout, this may introduce an $O(s^2)$ delay in the worst case for polysilicon interconnection (by Assumption 2.11). Thus for most practical purposes, it is neither cost-effective nor exactly feasible to transform any arbitrary rectangular layout into a square layout.

4.4. An Alternate Layout Strategy:

In order to tackle this issue of layout transformation, the problem can be viewed into from a different perspective. First of all, it is not entirely correct to assume that the optimal layout of the processing elements is exactly rectangular in shape. Since the processing element is comprised of functional blocks, like cpu, memory, i/o buffer, etc., they can be organized in different ways. Secondly, like the square blocks, many other regular geometrical shapes can be distributed uniformly to yield asymptotically similar delay both along the horizontal and the vertical directions of the chip. In this section, a set of geometrical shapes have been identified for the basic blocks. Thus instead of depending on post processing the optimal layout and transforming it into a square layout, a target shape is determined from a set of shapes which can be cost-effectively laid out on the chip. In rest of the thesis the layout geometries of the processors which

describe Cellular Array Networks only have been discussed. These networks have been shown (in Chapter 3) to be highly suitable in a VLSI environment and hence the motivation for identification of the layout geometries which can be cost-effectively embedded on the IC. For these networks having a nearest neighbor type array structure, different types of planar tiles which can be packed densely in the square area, can be used as basic blocks.

The constraint on the shape of the block is that it should be convex polygon (Assumption 2.1) and the incident interconnection wires should be perpendicular to the edges of the polygon. The conventional VLSI technologies use the Manhattan interconnection techniques, in which the interconnection wires run both vertically (in one layer) and horizontally (in another layer), parallel to the edges of the chip (Assumption 2.5). A class of tiles, called **Polyominoes,** satisfy this constraint very well and they can be arranged to cover a larger square area with maximum packing density. Polyomino tile consists of one or more unit squares; if the tile consists of n squares, it is called **n−omino.** The planar covering properties of these structures have been studied in detail by **Golomb** [Gol65] in connection to problems on recreational mathematics. Thus, the concept behind converting a rectangular layout to a square layout is a special case, viz., the property of planar covering by 1-omino (or monomino). The next chapter discusses the properties of lower order ($n \leq 6$) ominoes as to their planar covering and what type of network they can map on the planar area. The choice of the lower order ominoes ensures that propagation delay is approximately the same, both horizontally and vertically across the chip. In particular, it will be shown that by employing an $O(\sqrt{N})$ embedding algorithm, the **2−omino** can describe communication graphs of degree 6 and hence can be exploited for systolic algorithms. Also it will be shown that by using an $O(logN)$ embedding algorithm, the **3−omino** can optimally embed the two dimensional fault-tolerant mesh structure with near optimal redundant cells.

4.5. Conclusion:

The main contribution of this chapter is to propose algorithms which transform a rectangular layout into a square layout. The practical difficulties and disadvantages of these algorithms have been identified and finally an alternate strategy has been suggested.

Chapter 5

CELLULAR EMBEDDING AND NETWORKS

5.1. Introduction

One of the principal design criteria for VLSI design, is that the layout pattern should be regular and repetitive. This helps in lowering the layout cost [MeC80] and designing simpler computational algorithms. **Kung** and **Leiserson** [KuL79, Kun79] have designed a number of systolic algorithms for both one dimensional and two dimensional cellular arrays which have homogeneous communication geometries. This section discusses the processor geometries which can area efficiently construct these communication geometries. Also, it has been shown that regular array structures with added features which are highly relevant under a VLSI environment can be constructed by the layout techniques discussed in this section.

At first, a formal framework for the construction of layout algorithms has been made in section 5.2 introducing the notion of the Euclidean planar covering by polyomino tiles. The processors are replaced by the tiles and the properties of polyomino tiles have been utilized to select the processor geometry for a particular communication structure. To maintain the regularity, cellular embedding algorithms have been designed by transforming tiles on the Euclidean plane by applying *linear transformations*. Linear transformations have linear as well as rotational components. The linear component is not included here and the tiles are juxtaposed to one another to align their outer edges so that they can be enclosed into a minimum square area. The rotational component has been defined as an operator and the tiles are restricted to orient along a finite direction as permitted by the layout constraint of the

VLSI design (Assumption 2.5). In section 5.3, the embedding algorithms have been succinctly represented as the spatial adjacency of tiled regions which can be recursively constructed employing the operators as postulated by the embedding rules. The layout cost, structural complexity and the layout area for different embedding algorithms are also discussed. Cellular Networks described by these mosaic layouts have been identified in section 5.4. The relevance of redundancy in fault-tolerant mesh networks has been discussed. Finally, in section 5.5, a number of cell geometries have been identified for regular array networks. Section 5.6 concludes by recounting the basic results.

5.2. Formal Framework of Cellular Embedding

Let $\mathbf{E}^2$ be a two dimensional Euclidean plane and $\{x, y\}$ be the basis on $\mathbf{E}^2$ such that the angle subtended by x and y at the point of intersection, called the origin, O, is equal to $\pi/2$. Let $X=\{ i \mid i=0,1,2 \cdots \}$ be the set of points on the x-axis such that the distance between any two neighboring points is equal to δx. Let $Y=\{ j \mid j=0,1,2, \cdots \}$ be the set of points on the y-axis such that the distance between any two neighboring points is equal to $\delta y = \delta x$. Let V denote the cartesian product of X and Y such that $V=X \times Y=\{ (i, j) \mid i \in X \ \& \ j \in Y \}$.

The quadratic lattice graph G on $\mathbf{E}^2$ is defined by connecting all the ordered pairs of V such that the edges are parallel to either the x-axis or y-axis. The (i, j)-th cell on $\mathbf{E}^2$ is the area bounded by the edges on G connecting the quad ordered pairs $\{(i, j), (i+1, j), (i, j+1), (i+1, j+1)\}$ and is denoted by

$$cell\ (i, j) = x^i y^j.$$

Definition 5.1. **Tile:** A tile, τ_k, on $\mathbf{E}^2$ consists of k rookwise [Gol65] connected cells and is called a k-omino.

Definition 5.2. **Shape Polynomial of a tile:** The shape polynomial, $\mathbf{S}(\tau_k)$, of the tile, τ_k, whose lower left corner is on the origin, O, is given by

$$S(\tau_k) = \sum_{(i,j)\in E^2} \gamma_{ij}\, x^i y^j$$

where,

$$\gamma_{ij} = \begin{cases} 1 & \textit{iff for all } (x, y)\in \textit{cell } (i, j) => (x, y)\in \tau_k, \\ 0 & \textit{otherwise.} \end{cases}$$

The area of τ_k is equal to $|S(\tau_k)|$.

Definition 5.3. **Conjugate of the Shape Polynomial:** The conjugate of shape polynomial, $S(\tau_k)$, is defined as a shape polynomial, $S^*(\tau_k)$, obtained by permuting x and y elements [Bar82] of $S(\tau_k)$ such that

$$S^*(\tau_k) = \sum_{i=0}^{\infty} \mu_{ji}\, x^j\, y^i$$

where, $\mu_{ji} = \gamma_{ij}$.

Definition 5.4. **Operator:** An operator, Ψ_ζ, rotates a tile, τ_k, by an angle Ψ about one of its edges parallel to the ζ-axis and the new shape polynomial of τ_k is given by $\Psi_\zeta \bullet S(\tau_k)$.

Properties of Ψ_ζ:

(1) If $\Psi_{\zeta 1}$ and $\Psi_{\zeta 2}$ are two operators applied on τ_k, then

$$(\Psi_{\zeta 1}\Psi_{\zeta 2}) \bullet S(\tau_k) = \Psi_{\zeta 1} \bullet (\Psi_{\zeta 2} \bullet S(\tau_k)) = \Psi_{\zeta 2} \bullet (\Psi_{\zeta 1} \bullet S(\tau_k)).$$

(2) If $\Psi = 2m\pi$ (where $m \in \{1,2,...\}$), then $(2m\pi)_\zeta \bullet S(\tau_k) = S(\tau_k)$ and $\iota = (2m\pi)_\zeta$ is called the **Identity Operator.**

(3) If $\Psi = \phi$, then $\phi \bullet S(\tau_k) = \phi$ and $\Psi = \phi$ is called the **Null Operator.**

Since the interconnections in a VLSI layout run either vertically or horizontally parallel to the edges of the chip, the tiles are oriented such that their edges are always parallel to the basis $\{x, y\}$.

Definition 5.5. **Permissible Orientation:** The permissible orientation of a tile τ_k, on E^2 are due to the following operators applied in any arbitrary sequence:

$(m\pi)_x$, $(m\pi)_y$, $(m\pi)_z$ and λ_z, where $\lambda_z \bullet S(\tau_k) = \pi_y \bullet S^*(\tau_k)$ and z is orthogonal to E^2. The transformations due to $(m\pi)_x$ and $(m\pi)_y$ are called **reflections** and transformations due to $(m\pi)_z$ and λ_z are called **rotations.**

Theorem 5.1: *A τ_k has at most 8 distinct shape polynomials corresponding to all permissible orientations.*

Proof: Let $S(\tau_k)$ be the shape polynomial of τ_k. By applying the operators π_x, π_y, and π_z on $S(\tau_k)$, three more shape polynomials can be generated. By permuting the x and y elements, $S^*(\tau_k)$ can be derived and the above operators can be applied to get three more shape polynomials. Depending on the symmetry of τ_k about x, y or z axes, there will be 8, 4, 2 or 1 distinct shape polynomials. □

Figure 5.1 shows the shape polynomials of a tile, designated as P_5, corresponding

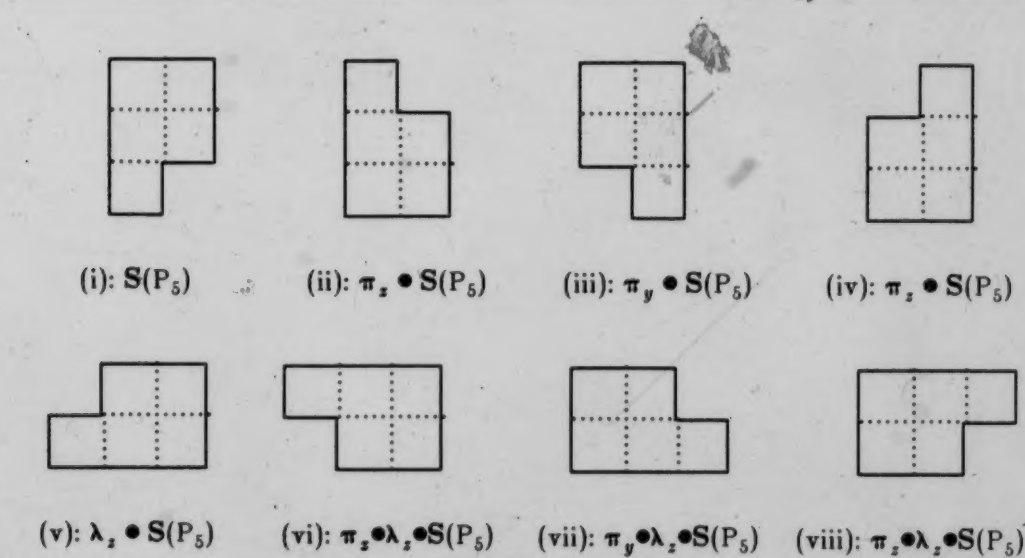


Figure 5.1 Permissible Orientation of P_5

to all its permissible orientations.

Definition 5.6. **Tile Notations:**

(1) A tile is denoted by I_k iff, by any permissible orientation, its shape polynomial can be represented by

$$S(I_k) = \sum_{i=0}^{k-1} x^i$$

Figure 5.2a shows I_2.

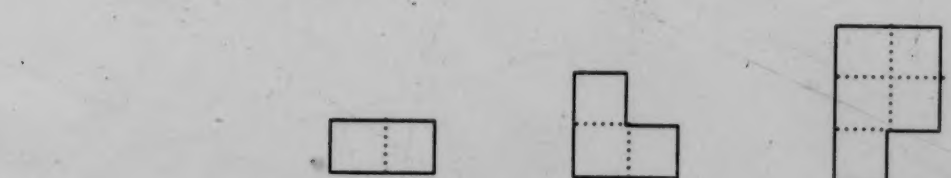
(2) A tile is denoted by L_k iff, by any permissible orientation, its shape polynomial can be represented by

$$S(L_k) = \sum_{i=0}^{m-1} x^i + \sum_{i=1}^{s} y^j, \qquad \text{s.t.} \quad m+s = k>2.$$

Figure 5.2b shows L_3.

(3) A tile is denoted by P_k iff, by any permissible orientation, the shape polynomial can be represented by

$$S(P_k) = \sum_{i=1}^{w} x^i \sum_{j=m}^{n-1} y^j + \sum_{j=0}^{m-1} y^j, \qquad \text{s.t.} \quad m+s(n-m)=k>2.$$



(a): I_2 (b): L_3 (c): P_5

Figure 5.2 Shape Polynomial of Some Polyominoes

Figure 5.2c shows P_5

In general, if by any permissible orientation, the shape polynomial of a tile approximately represents an English letter in block capital, τ, then the tile is represented by τ_k, where k is the number of cells in the tile.

Definition 5.7. **Cellular Embedding:** The cellular embedding of a region $R = \bigcup_{i=1}^{n} R_i$ is its shape polynomial $\mathbf{S}$, represented as a spatial distribution of the shape polynomials $\mathbf{S}_i$'s (or the permissible orientations) of R_i's.

Assuming two of the edges of R constitute the basis $\{x, y\}$ such that the lower left corner of R is the origin, the spatial distributions of $\mathbf{S}_i$'s can be expressed as a matrix, such that for all R_i and R_j, if R_j is adjacent to R_i horizontally, vertically or diagonally, $\mathbf{S}_i$ and $\mathbf{S}_j$ are also similarly adjacent in the matrix.

If $R = \bigcup_{i=1}^{9} R_i$ (Figure 5.3) is a planar region having Moore's neighborhood structure [Hay84], then

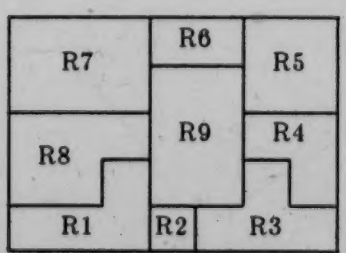


Figure 5.3 Planar Regions having Moore's Neighborhood

$$\mathbf{S} = \begin{pmatrix} S_7 & S_6 & S_5 \\ S_8 & S_9 & S_4 \\ S_1 & S_2 & S_3 \end{pmatrix} \tag{5.1}$$

The RHS is called the **Embedding Matrix** and the equation (5.1) is called the **Embedding Rule, ϵ**.

Operations on S:

The operations on **S** (where **S** is given by Eqn 5.1) are shown below:

(1) Reflection about x-axis

$$\pi_x \bullet \mathbf{S} = \begin{pmatrix} \pi_x \bullet S_1 & \pi_x \bullet S_2 & \pi_x \bullet S_3 \\ \pi_x \bullet S_8 & \pi_x \bullet S_9 & \pi_x \bullet S_4 \\ \pi_x \bullet S_7 & \pi_x \bullet S_6 & \pi_x \bullet S_5 \end{pmatrix}$$

(2) Reflection about y-axis

$$\pi_y \bullet \mathbf{S} = \begin{pmatrix} \pi_y \bullet S_5 & \pi_y \bullet S_6 & \pi_y \bullet S_7 \\ \pi_y \bullet S_4 & \pi_y \bullet S_9 & \pi_y \bullet S_8 \\ \pi_y \bullet S_3 & \pi_y \bullet S_2 & \pi_y \bullet S_1 \end{pmatrix}$$

(3) Rotation about z-axis

$$\pi_z \bullet \mathbf{S} = \begin{pmatrix} \pi_z \bullet S_3 & \pi_z \bullet S_2 & \pi_z \bullet S_1 \\ \pi_z \bullet S_4 & \pi_z \bullet S_9 & \pi_z \bullet S_8 \\ \pi_z \bullet S_5 & \pi_z \bullet S_6 & \pi_z \bullet S_7 \end{pmatrix}$$

(4) Conjugate operation

$$\lambda_z \bullet \mathbf{S} = \begin{pmatrix} \lambda_z \bullet S_5 & \lambda_z \bullet S_4 & \lambda_z \bullet S_3 \\ \lambda_z \bullet S_6 & \lambda_z \bullet S_9 & \lambda_z \bullet S_2 \\ \lambda_z \bullet S_7 & \lambda_z \bullet S_8 & \lambda_z \bullet S_1 \end{pmatrix}$$

5.3. Mosaic Layout Constructed by Polyomino Tiles:

The cellular embedding of a repetitive tile structure is called a **Mosaic.** For a VLSI implementation, such regularly structured layouts are very easy to construct. These layouts are constructed by replicating a singular type tile such that there is no gap between the successive tiles. Formally, a mosaic layout can be defined as

Definition 5.8. **Mosaic:** A layout of τ_k under the embedding rule ϵ is called a Mosaic, $\mathbf{M}_{\epsilon}(\tau_k)$, iff for every point within the layout, there exists an instance of the tile, τ_k denoted as τ_k^i, such that for all $j \neq i$

$$\tau_k^i \bigcap \tau_k^j = \phi$$

A mosaic layout constructed by L_3 and I_2 will be discussed here. For other polyomino structures, the layout algorithms can be easily constructed. The results of **Golomb's** [Gol66] work on planar covering of polyomino tiles can be utilized to construct these algorithms.

5.3.1. Embedding Algorithms

Theorem 5.2: *The layout of* L_3 *describes a mosaic under the following embedding rule (say* $\epsilon 1$*):*

$$\mathbf{S}_n(L_3) = \begin{pmatrix} \pi_x \bullet \mathbf{S}_{n-1}(L_3) & \phi & \mathbf{S}_{n-1}(L_3) \\ \phi & \mathbf{S}_1(L_3) & \phi \\ \mathbf{S}_{n-1}(L_3) & \phi & \pi_y \bullet \mathbf{S}_{n-1}(L_3) \end{pmatrix}$$

where, $n>1$ *is called the* **Level of Embedding** *and* $\mathbf{S}_1(L_3)=\mathbf{S}(L_3)$ *is the shape polynomial of* L_3.

Proof: Refer to Figure 5.4. In order that a tiled region having shape polynomial $\mathbf{S}_m(L_3)$ is a mosaic by embedding rule $\epsilon 1$, it is required that $h2=4^{m-1}h(L_3)-1$, $v2=4^{m-1}v(L_3)-1$, $h3=h(L_3)$ and $v3=v(L_3)$. For $n=2$, it can be observed that L_3 constructs a mosaic under embedding rule, $\epsilon 1$. By employing the induction method, the theorem can be easily proved. □

Theorem 5.3 *The layout of I_g describes a mosaic, under the following embedding rule (say* $\epsilon 2$*):*

$$S_n(I_2) = \begin{pmatrix} \phi & S(I_{2n}) & \phi \\ \lambda_x \bullet S(I_{2n-2}) & S_{n-1}(I_2) & \lambda_x \bullet S(I_{2n-2}) \\ \phi & S(I_{2n}) & \phi \end{pmatrix}$$

where $n \geq 1$ *is called the* **Level of Embedding** *and* $S_1(I_2) = S(I_2)$ *is the shape polynomial of* I_g. ∎

Proof: The n-th level of embedding consists of $4n-2$ dominoes, n dominoes in top row, n dominoes in bottom row $n-1$ dominoes in left-most column and $n-1$ dominoes in rightmost column. This describes an annular square region, having inner edge of length $n-1$ and outer edge of length n. By simple induction the theorem can be proved. □

5.3.2. Computational Power

The computational power of a mosaic layout due to an n level of embedding is measured by a parameter called **Structural Complexity.** Since each processor has

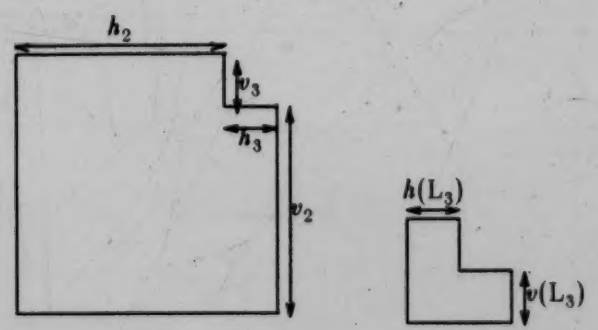


Figure 5.4 Mosaic Layout described by L_3

equal and constant computational power, a mosaic of an n level embedding can have exponential or polynomial computational power depending on its structural complexity.

Definition 10.9. **Structural Complexity:** The structural complexity of the shape polynomial $\mathbf{S}_n(\tau_t)$ is the total number of constituent k-ominoes.

Theorem 5.4: *The structural complexity of* $\mathbf{S}_n(L_3)$ *is given by*

$$N_n(L_3) = \frac{1}{3}(4^{n+1}-1)$$

Proof: From the recursive definition of $\epsilon 1$,

$$\mathbf{N}_n(L_3) = 4\mathbf{N}_{n-1}(L_3) + 1, \quad \text{with } \mathbf{N}_0(L_3) = 1.$$
$$\Rightarrow \mathbf{N}_n(L_3) = 4^n N_0(L_3) + \sum_{i=0}^{n-1} 4^i = \frac{1}{3}(4^{n+1}-1). \ \square$$

Theorem 5.5 *The structural complexity of* $\mathbf{S}_n(I_2)$ *is given by*

$$\mathbf{N}_n(I_2) = 2n^2.$$

Proof: From the recursive definition of $\epsilon 2$,

$$\mathbf{N}_n(I_2) = 4n + \mathbf{N}_{n-1}(I_2) - 2, \quad \text{with } \mathbf{N}_1(I_2) = 2.$$
$$\Rightarrow \mathbf{N}_n(I_2) = 4\sum_{i=2}^{n} i + \mathbf{N}_1(I_2) - 2(n-1) = 2n^2. \ \square$$

5.3.3. Chip Area

Definition 5.10. **Square Hull:** The square hull of a mosaic is the smallest size of square in which the mosaic can be enclosed.

Thus the square hull of a mosaic is a measure of the minimum chip size required to fabricate the mosaic layout. In order that the mosaic can be area efficiently embedded within a chip, the mosaic should be approximately square in shape. The mosaic $\mathbf{M}_{\epsilon 1}(L_3)$ has a square hull of side $2^{n+1}/\sqrt{3}$, assuming $|\mathbf{S}(L_3)| = 1$. The mosaic

$\mathbf{M}_{\epsilon 2}(I_2)$ has a square hull of side $\sqrt{2n}$.

5.3.4. Layout cost

Definition 5.11. **Layout Cost:** The layout cost is defined as the cost of embedding.

Theorem 5.6: *If the layout cost for placement of a tile of* L_3 *is* $O(1)$, *the mosaic* $\mathbf{M}_{\epsilon 1}(L_3)$ *having structural complexity* $\mathbf{N}_n(L_3)$ *can be constructed by paying a layout cost of* $O(log \mathbf{N}_n(L_3))$.

Proof: Each level of recursion of the embedding $\epsilon 1$ needs a cost of $5 \times O(1)$ and to construct $\mathbf{M}_{\epsilon 1}(L_3)$, a cost of $5n \times O(1)$ is necessary. Using Theorem 5.3, the total layout cost can be calculated to $5/2 \log (3\mathbf{N}_n(L_3)+1) = O(log\ \mathbf{N}_n(L_3))$. □

Theorem 5.7: *If the cost of placing of one tile* (I_2) *is* $O(1)$, *then the total cost of* $\mathbf{M}_{\epsilon 2}(I_2)$ *is equal to* $4(n-1)O(1) = O(\sqrt{\mathbf{N}_n(I_2)})$.

Proof: Each level of recursion of layout involves in adding the top and the bottom rows and the rightmost and the leftmost columns and needs $4 \times O(1)$ cost except when $n=1$, which needs $2 \times O(1)$ cost. Thus to construct $\mathbf{M}_{\epsilon 2}(I_2)$, it needs at most $(4n-2) \times O(1)$ cost. Applying Theorem 5.4, the total cost for constructing $\mathbf{M}_{\epsilon 2}(I_2)$, at most $O(\sqrt{\mathbf{N}_n(I_2)})$. □

5.4. Interconnection Networks described by Mosaic Layout:

In order to evaluate the importance of the embedding algorithms described earlier, it is necessary to ascertain the interconnection networks described by the mosaic patterns. Interconnection Networks can be called *Communication Graphs* in graph theoretic terms. In reference to the VLSI computational model described in Chapter 2,

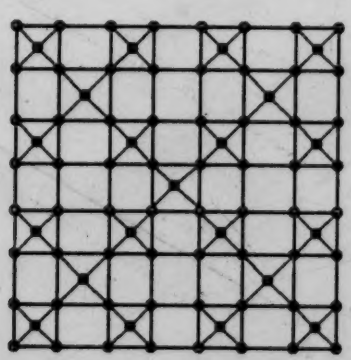
a communication graph is described as follow:

Definition 5.12. **Communication Graph:** A graph, $\mathbf{G}_{\epsilon}(\tau_k)$, is called the communication graph of τ_k under embedding rule ϵ, iff nodes in the graph denote the tiles and edges denote the communication path between the nodes.

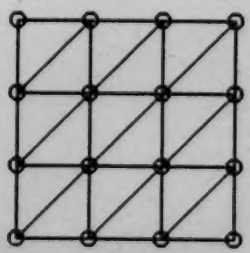
Definition 5.13. **Degree of Communication Graph:** The degree, $\delta(\tau_k)$, of a communication graph, $\mathbf{G}_{\epsilon}(\tau_k)$, is defined as the maximum number of incident edges at any node of $\mathbf{G}_{\epsilon}(\tau_k)$. A communication graph, $\mathbf{G}_{\epsilon}(\tau_k)$, is r-regular iff all its non-peripheral (i.e., internal) nodes have r incident edges.

The communication graphs described by $\mathbf{M}_{\epsilon 1}(L_3)$ and $\mathbf{M}_{\epsilon 2}(I_2)$ are shown in Figure 5.5. The graph due to $\mathbf{M}_{\epsilon 2}(\tau_k)$ is a 6-regular and corresponds to the communication geometries of hexagonal arrays [Kun79]. These arrays have been well exploited by many researchers to design cost-effective systolic algorithms for band matrix multiplication [KuL79, WiD81], LU-decomposition [MeC80], Fast Fourier Transformation [Tho83], 2-D convolution [KuS81], Reed-Solomon Encoding [Liu81], etc.

The graph described by $\mathbf{M}_{\epsilon 1}(L_3)$ denotes communication geometries of square arrays with redundant cells. This communication structure is highly suitable for fault-tolerant parallel processing and is particularly relevant for high yield VLSI fabrication. Approximately, 33% of redundant processors are available for 2-dimensional mesh operation. In order to justify the 33% redundancy, it can be noted that approximately 12% redundancy is required to maximize the yield in well-controlled processes like CMOS while approximately 30% redundancy required in processes having high defect density like in GaAs technology (Assumption 5.3). The remaining 20% redundancy available with CMOS ICs can be utilized to improve the operational reliability. For VLSI technology, because of device scaling at the submicron level a host of hostile phenomena like electromigration, Kirkendall effects, hot electron effects, radiation,



5.5a: Communication Graph for Fault-tolerant Meshes



5.5b: Communication Graph for Hexagonal Arrays

Figure 5.5 Communication Graphs described by Mosaic Layout

etc., are likely to impair the performance of the chips during normal operation. It is believed that to enhance the reliability of operation, frequent monitoring of the computational correctness should be made. This can be achieved by employing auto-corrector circuits but they usually need extra spaces and complicate the design of the processors. Thus the coding technique of fault-tolerance is not suitable for VLSI implementation. An alternate technique to improve the reliability is to simultaneously compute the same data by two or more processors and to ascertain the correctness. Since the redundant processors are symmetrically located in the mesh networks (Figure 5.5), they can check the correctness of the outer four processors in a round robin fashion

and thereby improve the operational reliability of the mesh networks considerably. Thus the mosaic layout, $\mathbf{M}_{\epsilon_1}(L_3)$ is topologically suitable for fault-tolerant computation. Regular mesh structure is not suitable for fault-tolerant computation. **Leiserson** and **Leighton** [LeL82] have investigated the problem of improving the yield of mesh networks. They have employed the *divide and conquer* paradigm to construct the mesh network by bypassing the *dead* cells. Assuming random occurrence of defects (Assumption 5.1), in the worst case, it needs $O(\log N \log\log N)$ wire length to connect two successive *live* cells in an N cell mesh. This introduces high delay (Theorem 2.1), creating a bottleneck in the systolic computation.

5.5. Cell geometries for Cellular Networks:

At this stage, it is interesting to note that these Cellular Networks can be described by a number of other polyomino tiles. It is not necessary to provide the layout algorithms for all these possible tile shapes. The planar covering properties of these tiles have been pictorially represented by **Golomb** [Gol66] and from these geometrical arrangements, the layout algorithms can be easily designed employing the concept developed in this chapter. Here, only the polyomino tiles have been identified which can be utilized to describe different Cellular Networks. The choice of lower order polyominoes has been restricted by the requirement of distributing the processors uniformly both horizontally and vertically so that delay is approximately same along both direction. Only the Cellular Networks of degree 4 and 6 are identified, because only they are suitable for practical algorithms [Kun79]. Table 5.1 identifies the layout geometries which can describe the Cellular Networks which are highly suitable for VLSI parallel processing.

Table 5.1 Layout Geometries for Cellular Networks

Tile Notation	Shape Polynomial	Degree of Graph
I_2	$1+x$	6
L_3	$1+x+y$	4 (Fault-tolerant)
T_4	$1+x+x^2+xy$	6
Z_4	$1+x+xy+x^2y$	6
F_5	$1+x+xy+xy^2+x^2y$	6
Y_5	$1+x+x^2+x^3+x^2y$	6
P_6	$1+y+y^2+y^3+x+xy$	4 (Fault-tolerant)

5.6. Conclusion:

The main result of this chapter is to identify a number of polyomino tile shapes which can be cost-effectively mapped on a VLSI chip to yield Cellular Networks. The embedding algorithms have been designed, the chip area has been calculated, the layout cost has been estimated, and finally, the Cellular Networks generated by these tiles have been identified. Specifically, the fault-tolerant meshes with redundant cells have been pointed out and its suitability in VLSI environment has been emphasized. Thus, the main contribution of this chapter is to identify a multitude of layout geometries of the processors and to break the myth of the *square shape*.

The concept developed in this chapter has been employed to design the planar topologies and tessellation matrices which recursively decompose the Euclidean plane to represent a quadtree data structure.

Mazumder, P. and Tartar, J. : Planar Topologies for Quadtreee Representation, *Congressus Numerantium 36 (1984), Utilitas Mathematica Publishing Incorporated*, Winnipeg, Canada.

Chapter 6

CONCLUSION

This chapter draws the conclusion of the thesis by summarizing the results of its investigation, by enumerating its specific contributions and finally by outlining the areas of future research.

6.1. Summary:

This thesis mainly attempts to re-assess the interconnection networks in the perspective of on-chip parallel processing, a timely venture in the light of VLSI technology. Interconnection networks have been segregated into topological equivalent classes and a representative of each class has been evaluated to reveal the behavior of the class. The evaluation criteria form a three dimensional model providing an insight of the area, speed and the cost of the integrated circuit. The analysis has been done asymptotically and a formal computational model has been proposed on the basis of the state-of-the-art CMOS technology with two levels of interconnection. From the comparison of performance, it has been noted that the Cellular Networks are most suitable for VLSI embedding.

The thesis also notes that the geometry of the Cellular Networks plays an important role on the overall performance of the networks. It makes a detailed investigation on the layout geometry of the processors. It has developed the algorithms for transforming a rectangular layout of arbitrary aspect ratio into a square layout. The choice of the square layout has been automatically dictated by the fact that small square blocks can be cost-effectively packed into a larger square chip area. The thesis investigates into the difficulties and disadvantages of such layout post-processing.

It has suggested an alternative strategy of pre-determining the processor geometry from a set of feasible shapes. Polyomino cellular shapes have been identified for cost-effective VLSI embedding of various Cellular Networks. The embedding algorithms have been designed, layout area and cost have been computed and the computational power of the networks has been estimated.

The results of the above work can be summarized as:

(1) Cellular Networks with small interconnection structure are the most suitable for VLSI parallel processing. These networks need small chip area, consume small power, have high chip yield, high layout regularity and very good fault-tolerance capability. The only disadvantage is that the delay is moderately high and thereby the computational speed is moderately low.

(2) In principle, it is possible to transform a rectangular layout of any arbitrary aspect ratio into a square layout. If the layout is broken along its longitudinal direction, unbounded skewed delay is introduced along the transversal direction of the layout. This delay can be bounded by a constant factor, if the bi-directional slicing is done to yield small square blocks. The overall expansion is always smaller than a factor of 3. The edges of the rectangle can be folded into the square layout and thereby an $O(A_S)$ delay can be introduced in the worst case, where A_S is the area of the chip.

(3) Polyomino cellular shapes can be utilized to select the physical geometry of the processing cells in the Cellular Networks. More than one type of cellular shapes are available for Cellular Networks of degree 4 and 6.

(4) Cellular Networks with additional features can be generated by the polyomino cells. Especially, two dimensional meshes with fault-tolerance capabilities can be cost-effectively made using the tromino and hexomino cellular structures. The redunandant cells are located symmetrically within the networks and can be employed to ameliorate the operational reliability of the networks.

6.2. Contribution of the thesis:

The main contributions of the thesis can be enumerated as follows:

(1) The thesis has proposed a formal computational model which represent all the aspects of MOS VLSI technology. The model can also be readily used for GaAs MESFET technology. The model explores the five aspects of the VLSI design - the technological, the embedding, the timing, the power consumption and the failure occurrences. This is a unique model in the literature.

(2) The thesis has conceptuallized the whole gamut of interconnection networks into four topologically equivalent classes. It has identified the representatives of each class and by perusing the behavior of each class, it has identified the class of networks which are most suitable in the VLSI design.

(3) The thesis has enumerated the criteria for performance evaluation under three orthogonal classes - the physical aspects, the computational aspects and the cost aspects. From the results of evaluation, it has been confirmed that the Cellular Networks are the most suitable networks for VLSI embedding. Such networks have been widely used by many researchers for systolic computation, because their physical regularity provides the basis of regular data flow. This thesis has explored these networks from additional viewpoints and further emphasized their suitability for VLSI design.

(4) The thesis has discussed the techniques to transform a natural rectangular layout of arbitrary dimension into a square layout.

(5) The thesis has identified the processor geometries which can be cost-effectively laid on the chip to construct the Cellular Networks.

(6) The thesis has identified the new class of Cellular Networks which are highly relevant in VLSI computation. It has specifically identified the fault-tolerant meshes which provide better chip yield and operational reliability.

6.3. Recommendation of Future Research:

The theoretical investigations done in this thesis have unearthed some promising areas for future research. A few of them are identified here:

6.3.1. Extension of Computational Model:

The computational model described in this chapter uses MOSFET technology. Similar computational models can be developed for other technologies like bipolar, MESFET, Josephson Junction Devices, etc.

The current model has not considered the effect of regularity of the interconnects on the chip yield. The randomness of interconnects pose additional problems in placing the redundant components and increases the chip area considerably. The topology of the networks should be taken into consideration to calculate the additional area required to lay the redundant processors.

6.3.2. Evaluation under Multi-level Interconnect Model:

In the computational model, it has been assumed that at best two level of interconnects are possible and only rectilinear interconnections are allowed. The networks have been evaluated under these assumptions. If the model is extended to three levels

of interconnects and restricted angular interconnections are allowed, then the area and the length of interconnects in networks like the CCC, the PSN, the OTN, etc., will be considerably less. Presently, many integrated circuit manufacturers are attempting to develop two level metal interconnects using gold-platinum-titanium-aluminum composite films. Gold and aluminum are the metal interconnects while the sandwiching materials like platinum and titanium are used to form insulating base between two levels of conducting materials. Allowance of fixed angular connections reduces the wiring complexity in many circumstances and many manufacturers are trying to introduce 45° angular bending. A detailed theoretical analysis to this effect will reveal the ultimate application potentiality of CCC, PSN, OTN, etc., which are very popular in non-VLSI parallel computation.

6.3.3. Exploration of other Cellular Geometries:

The extension of computational models to multi-level interconnects with angular bending will allow to explore for other regular structures like regular polygons, poliamonds, polihexes, etc., for cellular geometries. Thus, the results of the combinatorial analysis of planar covering properties of different classes of tiles, can be utilized to select the layout of the processors in automated layout generation.

Bibliography

[AbA80] H. Abelson and P. Andreae, Information transfer and area-time tradeoffs for VLSI multiplication , *Comm. ACM 23*, 1 (Jan. 1980), 20-23.

[AlR80] R. Aleliunas and A. L. Rosenberg, On embedding rectangular grids in square grids, Technical Report, University of Toronto, 1980.

[AnJ75] G. A. Andersen and E. D. Jensen, Computer interconnection structures: taxonomy, characteristics and examples, *ACM Computer Survey 7*, (Dec. 1975), 197-213.

[Ata83] M. J. Atallah, *Algorithms for VLSI networks of processors*, PhD Thesis, The Johns Hopkins University, 1983.

[Bar80] D. F. Barbe, *Very Large Scale Integration: Fundamentals and Applications*, Springer Verlag, 1980.

[Bar82] F. W. Barnes, Algebraic theory of brick packing 2, *Discrete Mathematics 42*, (1982), 129-144.

[BPP82] G. Bilardi, M. Pracchi and F. P. Preparata, A critique of Network Speed in VLSI Models of Computation, *IEEE Journal of Solid-State Circuits SC-17*, 4 (Aug. 1982), 696-702.

[BrK80] R. P. Brent and H. T. Kung, The chip complexity of binary arithmetic, *Proceedings of 12th Annual ACM Symp. on Theory of Computing*, 1980, 190-200.

[BrG82] R. P. Brent and L. M. Goldschlager, Some area time tradeoffs for VLSI , *SIAMJC 11*, 4 (Apr. 1982), 737-747.

[Bur83] D. Bursky, 1983 technology forecast - digital LSI, *Electronic Design*, Jan. 1983, 103-128.

[CaS81] P. R. Cappello and K. Steiglitz, Area-efficient VLSI structures for multiplying at clock rate, Technical Report #289, Princeton University , 1981.

[ChM81] B. M. Chazelle and L. M. Monier, Model of computation for VLSI with related complexity results, *Proceedings of the 13th Annual ACM Symp. on Theory of Computing*, 1981, 318-325.

[CuS78] M. Cutler and Y. Shiloach, Permutation layout, *Networks 8*, (1978), 253-278.

[DeP79] A. Despian and D. Patterson, X-tree: A structured multiprocessor computer architecture, *Proceedings of IEEE 6th Annual Symposium on Computer Architectures*, Apr. 1979, 83-89.

[Dev80] Solid State Devices, , 1980.

[Don79] W. E. Donath, Placement and average interconnection lengths of computer logic, *IEEE Transaction on Circuits and Systems CAS-26*, 4 (Apr. 1979), 272-277.

[Elm77] B. R. Elmer, Fault-tolerant 92160 bit multiphase CCD memory, *ISCCC*, Feb. 1977, 116-117.

[GJS74] M. R. Garey, D. S. Johnson and L. Stockmeyer, Some simplified polynomial complete problems, *Proceedings of the 6th Annual ACM Symp. on Theory of Computing*, 1974, 47-63.

[Gha83] S. Ghandhi, *VLSI fabrication principles: Silicon and Gallelium Arsenide*, John Wiley & Sons, New York, NY, 1983.

[Gol65] S. W. Golomb, *Polyominoes*, Scribner's, 1965.

[Gol66] S. W. Golomb, Tiling with Polyominoes, *Journal of Combinatorial Theory 2*, 1 (1966), 280-296.

[Hay84] B. Hayes, Computer Recreations, *Scientific American 250*, 3 (1984), 12-21.

[HoZ81] E. Horowitz and A. Zorat, The binary tree as an interconnection network: applications to multiprocessor systems and VLSI, *IEEE Trans. on Computers c-30*, 4 (Apr. 1981), 247-253.

[KuL79] H. T. Kung and C. E. Leiserson, Systolic arrays (for VLSI), CMU-CS-79-103, Carnegie-Mellon University, Apr. 1979.

[Kun79] H. T. Kung, The structure of parallel algorithms, CMU-CS-79-143, Carnegie-Mellon University , Aug. 1979.

[KuS81] H. T. Kung and S. W. Song, A systolic 2-D convolution chip, CMU-CS-81-110, Carnegie-Mellon University, Mar. 1981.

[Lee61] C. Y. Lee, An algorithm for path connection and its applications, *IRE Transactions on Electronic Computers EC-10*, 3 (Sep. 1961), 346-365.

[Lei81] F. T. Leighton , New lower bound techniques for VLSI , *Proceedings of 22nd Annual IEEE Symposium on Foundations of Computer Science*, 1981, 1-12 .

[Lei82] F. T. Leighton , A layout strategy for VLSI which is provably good, *Proceedings 14th Annual ACM Symp. on Theory of Computing*, 1982, 85-98.

[LeL82] F. T. Leighton and C. E. Leiserson, Wafer-scale integration of systolic arrays, *Proceedings of 23rd Annual IEEE Symposium on Foundation of Computer Science*, 1982, 297-311.

[Lei81] C. E. Leiserson, *Area-Efficient grph layouts (for VLSI)*, PhD Thesis, Carnegie-Mellon University, 1981.

[LiT80] R. J. Lipton and R. E. Tarjan, A planar seperator theorem, *SIAM Journal on Applied Mathematics 36*, 2 (1980), 177-189.

[LiS81] R. J. Lipton and R. Sedgewick, Lower bounds for VLSI, *Proceedings of 13th Annual ACM Symp. on Theory of Computing*, 1981, 300-307.

[Liu81] K. Y. Liu, Architecture for VLSI design of Reed-Solomon encoders, *Proceedings of 2nd Caltech Conference on VLSI*, Jan. 1981, 539-553.

[Man81] T. E. Mangir, *Fault-tolerant design for VLSI design: Effect of interconnect requirements on yield improvement of VLSI design*, PhD Thesis, University of California, Los Angeles, 1981.

[MaA71] G. E. Marihugh and R. E. Anderson, The H diagram: a graphical approach to logic design, *IEEE Trans. on Computers*, 1971, 1192-1196.

[MeC80] C. A. Mead and L. A. Conway, *Introduction to VLSI systems*, Addison Wesley, Reading, MA, 1980.

[MeR82] C. Mead and M. Rem, Minimum propagation delays in VLSI, *IEEE Journal of Solid State Circuits SC-17*, 4 (Aug. 1982), 773-775.

[Moo70] G. E. Moore, What level of integration is best for you?, *Electronics*, Feb. 1970, 126-130.

[MoR76] M. Moshell and J. Rothstein, Parallel recognition of patterns: insights from formal language theory, *Proceedings of International Conference on Parallel Processing*, Aug. 1976, 222-229.

[Mur64] B. T. Murphy, Cost-size optima of monolithic integrated circuits, *Proceedings of IEEE*, Dec. 1964, 1537-1545.

[NMB83] D. Nath, S. N. Maheswari and P. C. P. Bhat, Efficient VLSI networks for parallel processing based on orthogonal trees, *IEEE Trans. on Computers c-32*, 6 (June 1983), 569-581.

[PrV79] F. P. Preparata and J. Vuillemin , The Cube Connected Cycles: A versatile network for parallel computation, *Proceedings of 20th Annual IEEE Symposium on Foundations of Computer Science* , 1979, 140-147.

[Pre82] F. P. Preparata, Three layers are enough, *Proceedings of 23rd Annual IEEE Symposium on Foundations of Computer Science* , 1982.

[Ram82] V. Ramachandran, Driving many long parallel wires , *Proceedings of 23th Annual IEEE Symposium on Foundations of Computer Science*, 1982, 369-378.

[Ros81] A. L. Rosenberg, Three-dimensional integrated circuitry, *Proceedings of the CMU Conference on VLSI Systems and Computations*, 1981, 69-80.

[SaM82] K. C. Saraswat and F. Mahammadi, Effect of scaling of interconnections on the time delay of VLSI circuit, *IEEE Journal of Solid State Circuits SC-17*, 2 (Apr. 1982), 275-280.

[Sav81] J. E. Savage, Planar circuit complexity and the performance of VLSI algorithms, *Proceedings of the CMU Conference on VLSI Systems and Computations*, 1981, 61-67.

[See67] R. B. Seeds, Yield, economic and logistic models for complex digital arrays, *IEEE Int. Convention Rec.*, Mar. 1967, 60-61.

[Sei80] C. L. Seitz, Self-timed VLSI systems, *Proceedings of CALTECH Conference of VLSI*, Jan. 1980, 345-355.

[Seq83] C. H. Sequin, Managing VLSI complexity: an outlook, *Proceedings of the IEEE 71*, 1 (1983), 149-166.

[Sie79] H. J. Siegel, A model of SIMD machines and a comparison of various interconnection networks, *IEEE Trans. on Computers c-28*, 12 (Dec. 1979), 907-917.

[Sta83] C. H. Stapper, Modeling of integrated circuit defect sensitivities, *IBM Journal of research and development 27*, 6 (June 1983), 549-557.

[StR80] D. Steinberg and M. Rodeh, A layout for the shuffle-exchange network with $O(N^2/\log^{3/2}N)$ area, Tech. Rep. 088 , IBM Scientific Center, Haifa, Israel, 1980.

[SuO73] I. E. Sutherland and D. Oestreicher, How big should a printed circuit board be?, *IEEE Trans. on Computers C-22*, 5 (May 1973), 537-542.

[Tho80] C. D. Thompson, *A complexity theory for VLSI* , PhD Thesis, Carnegie-Mellon University, 1980.

[Tho83] C. D. Thompson, The VLSI complexity of sorting , *IEEE Trans. on Computers C-32*, 12 (1983), 1171-1184.

[Thu74] K. J. Thurber, Interconnection networks - A survey and assessment, *AFIPS Conference Proceedings 43*, (1974), 909-919.

[Ull84] J. D. Ullman, *Computational Aspects of VLSI*, Computer Science Press, Woodland Hills, CA, 1984.

[Val81] L. G. Valiant , Universality considerations of VLSI circuits, *IEEE Trans. on Computers c-30*, 2 (Feb. 1981), 135-140.

[Vui80] J. E. Vuillemin , A combinatorial limit to the computing power of VLSI circuits, *Proceedings of 21st Annual IEEE Symposium on Foundations of Computer Science*, 1980, 294-300.

[WiD81] U. Wieser and A. Davis, A wavefront notation tool for VLSI array design, *Proceedings of CMU conference on VLSI Systems and Computations*, Oct. 1981, 226-234.

[Wit81] L. D. Wittie, Communications structures for large networks of microcomputers, *IEEE Trans. on Computers c-30*, 4 (Apr. 1981), 264-273.

[Yao79] A. C. Yao, Some complexity questions related to distributive computing, *Proceedings of 11th Annual ACM Symp. on Theory of Computing*, 1979, 209-213.

[Yao81] A. C. Yao, The entropic limitations of VLSI computations, *Proceedings of 13th Annual ACM Symp. on Theory of Computing*, 1981, 308-311.